COMPUTER MODEL OF A PASSIVE SYNTHETIC

APERTURE IMAGING SYSTEM

THESIS

Christopher P. Kane
Captain, USAF

AFIT/GEO/ENP/85D-2





DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GEO/ENP/85



COMPUTER MODEL OF A PASSIVE SYNTHETIC

APERTURE IMAGING SYSTEM

THESIS

Christopher P. Kane
Captain, USAF

AFIT/GEO/ENP/85D-2

Approved for public release; distribution unlimited

AFIT/GEO/ENP/85D-2

# COMPUTER MODEL OF A PASSIVE SYNTHETIC APERTURE IMAGING SYSTEM

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Electrical Engineering


Christopher P. Kane

Captain, USAF


December 1985



Approved for public release; distribution unlimited

## Acknowledgements

I would like to take time here to thank all those who helped me in this grand undertaking. I would first of all like to thank my thesis advisor, Maj Jim Mills, for not only guiding me but putting up with my occasional bouts of "mental blindness". I am also deeply indebted to the men and women of the Electro-Optical Sensors Laboratory. I am especially thankful to one member, Jeff Sweet, for his great help and advice in using their computer facilities and for letting me use the facilities in the first place.

Finally, I would like take a moment to thank one person that is very special to me. That person would be, of course, my wife, Sue. She has managed to put up with long lonely weekends and long lonely days while I have been slaving away behind a computer graphics terminal. I hope to make it up to her soon. All I can say now is that I love her more now than ever.

Christopher P. Kane, Capt, USAF

GEO-85D

## Table of Contents

## List of Figures

AFIT/GEO/ENP/85D-2

## Abstract

This thesis was concerned with the development of a computer model of a passive synthetic aperture imaging system. The research was divided into three parts. They were (1) applying an understanding of partial coherence theory and its relationship to the impulse response of the system, (2) developing the computer model, and (3) exercising the computer model to perform a sensitivity analysis.

The system modeled consisted of two lenses mounted on a movable platform. The lenses were separated by a fixed distance and travelled in a direction parallel to this separation. The coherence of radiation present at each lens emanating from a real source was measured yielding the Fourier transform of the source intensity distribution according to the van Cittert-Zernike theorem (2:510). The transform was then multiplied by an effective aperture (obtained from the motion and position of the lenses relative to the source). An inverse Fourier transform was then applied to this result yielding the image. This is the process modeled by the computer.

The results indicated that new means of image interpretation must be developed in order to make the results useful. This is due to the fact that the system behaves much like a high pass filter and the image is edge enhanced and not a scaled version of the geometric image.

# I. Introduction

The goal of this research was to perform a sensitivity analysis of the imaging performance of a passive interferometric imaging system. A hypothetical system consisting of two lenses that are physically connected yet separated by a fixed distance was determined to be the simplest case. Such a system could be mounted on a movable platform. The system samples the partially coherent radiation emanating from a source as its field-of-view travels across the source.

A cross-correlation is performed between the radiation fields present at each lens at predetermined intervals of time. This results in a set of discrete samples of the Fourier transform of the source radiant intensity distribution being measured (4:2-1). The source radiant intensity distribution may then be found by taking the inverse Fourier transform. This is a direct application of the van Cittert-Zernike theorem (2:510).

The technique of passive interferometric imaging is not new. Radio astronomers have used it for quite some time to obtain the angular diameter and brightness distributions of celestial bodies (13:2115). Efforts to implement this technique at optical or infrared frequencies have yielded promising results (14:1). Current synthetic aperture systems require the transmission of a coherent signal to obtain an image (4:2-1). Attention is now being given to passive interferometric techniques in the infrared region because this technique does not require the use of an active illumination which could reveal the detector's presence. High resolution has been obtained in the radio frequency region (13:2114) and it is desired to see if this can be duplicated at infrared frequencies.

## Goal of the Thesis

The goal of this thesis was to develop the necessary background and methods to perform a sensitivity analysis and to use this information to form a computer model of the system. This was done by constructing a model of the overall system. The computer model was then exercised to see how varying operating conditions and parameters that the imaging performance is dependent upon affected system performance. These results were compared to the results obtained under what were defined as ideal operating conditions to see what the effects were.

The thesis was divided into three phases in order to meet these broad objectives. They were (1) applying an understanding of partial coherence theory and its relationship to the impulse response of the system, (2) developing the computer model, and (3) exercising the computer model to perform the sensitivity analysis. The details of these phases are enumerated below.

Phase One. Understanding partial coherence theory revolves around an understanding of the propagation of the mutual intensity function (1:31). This is the quantity measured by the system. A normalized version of the mutual intensity function is related to the Fourier transform of the source intensity distribution. The image is then found by taking the inverse Fourier transform of the detected normalized mutual intensity function. Understanding partial coherence and the mutual intensity function and how it is measured are therefore the first important steps in analyzing the problem.

Understanding the contribution of the van Cittert-Zernike theorem is the next step in obtaining the impulse response of the system. As is stated in this theorem, it is assumed that the source is spatially incoherent and emits quasi-monochromatic light. The system behaves somewhat like a coherent imaging system despite the source being incoherent. This is illustrated in the following paragraph.

It can be shown (7:110-113) that for a given object amplitude distribution $u(X_o)$, the resultant coherent image amplitude distribution $u(X_i)$ is given by

$$u(X_i) = u(X_o) * h(X) \tag{1.1}$$

where * denotes a convolution and h(X) is the amplitude impulse response of the imaging system.

The amplitude impulse response h(X) is the Fourier transform of the pupil function $p(\lambda dx)$ (7:105) where $\lambda$ is the wavelength and d is the distance from the exit pupil of the optical system to the image plane. Taking the Fourier transform of both sides of Eq (1.1) yields the linear system equation

$$U(f_i) = U(f_o)P(\lambda df) \tag{1.2}$$

where $U(f_i)$, $U(f_o)$, and $P(\lambda df)$ are the Fourier transforms of $u(X_i)$, $u(X_o)$, and $p(\lambda dx)$ respectively.

This shows that the Fourier transform of the image amplitude distribution is directly proportional to the Fourier transform of the object amplitude distribution. The proportionality constant is the scaled pupil function. As noted earlier, the van Cittert-Zernike theorem states that the mutual intensity function is the Fourier

transform of the source intensity distribution. Therefore, applying the van Cittert-Zernike theorem to a given source intensity distribution yields the input to the linear system denoted in Eq (1.2). The transfer function, $P(\lambda df)$, is then the only parameter needed to determine the image amplitude distribution. Once $P(\lambda df)$ has been determined, $U(f_i)$ can be calculated for any source amplitude distribution (via Eq (1.2)) and $u(X_i)$ can be found by taking the inverse Fourier transform of $U(f_i)$.

This first phase identified the parameters and conditions which may effect the impulse response and system performance. The parameters initially identified were the distance between the lenses, aperture size, aperture fill, and frequency spacing or sample spacing.

Phase Two. The second phase was the development and testing of the computer model. The scenario upon which the model was based is as follows. A collection platform has two lenses mounted on it. These lenses are identical and are separated by a fixed distance. The path travelled by the collector is parallel to this separation. The slant range to the target is large enough such that it lies in the same plane as the target to be imaged. The collector traverses an angle $\Delta\theta$ as it moves past the target. This scenario is depicted in Figure 1.1.

The collector ideally would be able to gather data along $\Delta\theta$ equal from 0 to 180 degrees. The mutual intensity function would be sampled all along this interval with a diffraction grating providing the means of collecting several frequencies each time a sample is taken. A point source was considered the simplest case since this would provide the

impulse response of the system. Deriving the images due to other types of sources can be found from this. The angle $\Delta\theta$ was the primary variable modeled.

This required the definition of the impulse response of the system. The results of the first phase provided the information necessary to define the impulse response. This function was the most important part of the model. The model was written in FORTRAN 9000 and was developed on an HP 9000 computer. The model was written in a manner that will enable a person of limited computer background to be able to use the program without having to study a long and complicated list of procedures.

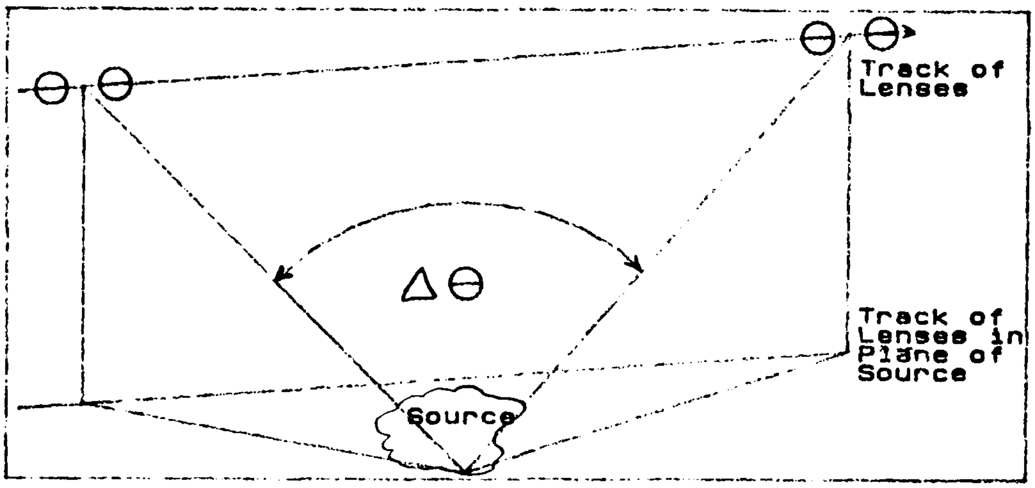



Fig 1.1. Envisioned Scenario of System Operation

Phase Three. The last phase consisted of exercising the computer model to obtain point spread functions of the system as well as images of certain simple objects. This was done by altering the ideal model of the second phase with changes to parameters identified in the first phase. The goal was to determine the system's imaging performance under realistic operating conditions.

# II. THEORY OF OPERATION

This chapter will better describe and more fully explain the theoretical basis of the optical system introduced in the previous chapter. It is assumed that the reader has an understanding of geometrical and Fourier optics. The chapter is divided into three sections. These are a physical and conceptual description of the optical system being modeled, the theoretical basis of operation, and linking the theory and the idea of a coherent imaging system to the optical system being modeled.

## The Optical System

The optical system consists of two main components. One is a pair of lenses separated by a distance d. The other is a linear array of detectors. The goal is to produce an image of a thermal source which emits a randomly fluctuating field by measuring the complex degree of partial coherence of the radiation field present at the lenses. The system can be conceived as one mounted on a movable platform that enables the system to rotate around the scene of interest. This is conceptually the same as letting the system remain stationary and allowing the scene to rotate below as shown in Fig. 2.1. Appendix A relates this simple geometry to the more complicated geometry of the moving lenses. Enough data must be collected so that an image of sufficient resolution and quality may be obtained by taking the two-dimensional inverse Fourier transform of the collected data.

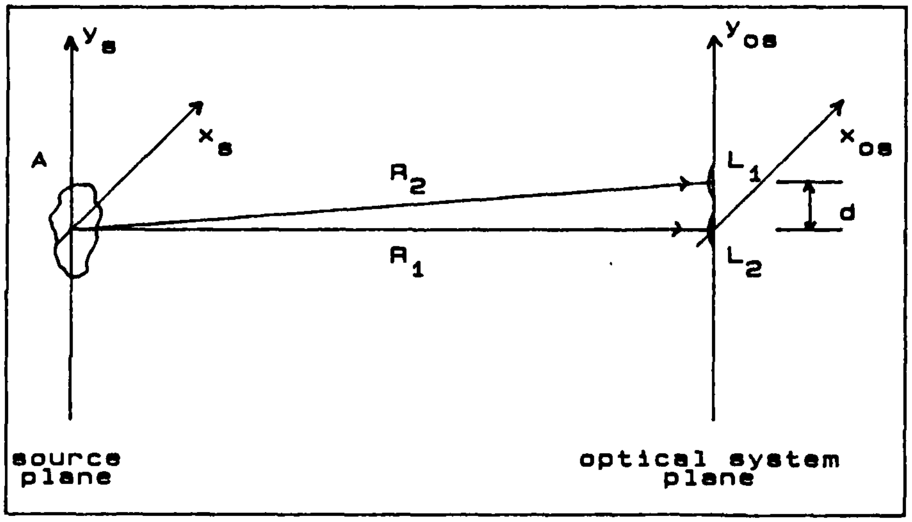



Fig 2.1. System Geometry

The scene A is assumed to be within the field of view of the system, $L_1$ and $L_2$ are the two lenses of the system, d is the distance by which the lenses are separated, $R_1$ and $R_2$ the distances from the center of the field of view to $L_1$ and $L_2$ respectively.

## Theoretical Basis of Operation

The simplest point from which to begin is Young's two slit experiment (1:7-11) which will be used to introduce the mutual intensity function and the complex degree of coherence. Consider the one-dimensional setup illustrated in Figure 2.2, where S is an extended polychromatic source in the X plane.

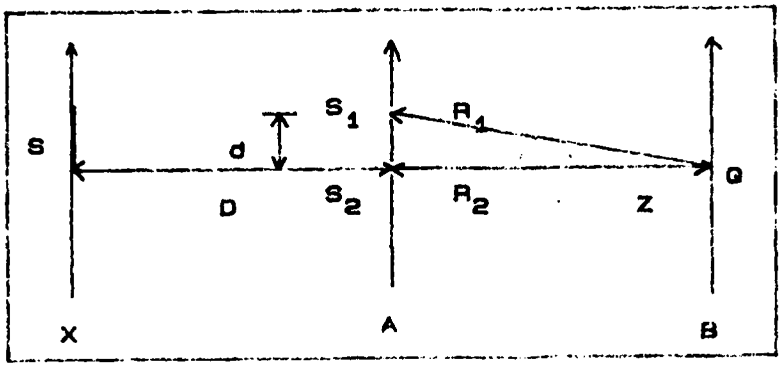



Fig 2.2. Young's Two Slit Experiment

$S_1$ and $S_2$ are identical slits separated by a distance d in the A plane which is otherwise opaque and is parallel to and a distance D from the X plane. Q is a point in the B plane which is parallel to and a distance Z from the A plane. $R_1$ and $R_2$ are the distances from $S_1$ and $S_2$ respectively to Q. A scalar treatment is sufficient here because the angles involved are assumed to satisfy the small angle approximation ($\sin\theta \cong \theta$).

The source S emits a complex incoherent light disturbance E(P,t). This disturbance propagates from the source to the plane A according to the wave equation (1:9)

$$(\nabla^2)E = (1/c)^2 \frac{d^2E}{dt} \qquad (2.1)$$

where c is the speed of light. The amplitudes of the disturbance at the

openings $S_1$ and $S_2$ are then denoted $E_1(t)$ and $E_2(t)$ respectively. The total disturbance at Q is then

$$EQ(t) = E_1(t-R_1/c) + E_2(t-R_2/c) \tag{2.2}$$

where $R_1/c$ and $R_2/c$ represent the time delays of $E_1(t)$ and $E_2(t)$ respectively in propagating to Q. Therefore, let $R_1/c = t_1$ and $R_2/c = t_2$. Rewriting Eq (2.2) with these changes yields

$$EQ(t) = E_1(t-t_1) + E_2(t-t_2) \tag{2.3}$$

The irradiance at Q, IQ, is necessarily a long time average of EQ(t). This is due to the fact that the frequency of the radiation field being sampled (EQ(t)) far exceeds the capability of the detectors employed to detect each individual oscillation. IQ is defined as

$$IQ = \langle(E_1(t-t_1) + E_2(t-t_2)\ (E_1(t-t_1) + E_2(t-t_2))^*\rangle \tag{2.4}$$

where <...> indicates the long time average of the quantity they enclose; i.e.

$$\langle I(t)\rangle = \lim_{T\to\infty} (1/T) \int_{-0}^{T} I(t)dt \tag{2.5}$$

<u>Mutual Coherence Function</u>. Carrying out the operations indicated in Eq (2.4) results in

$$IQ = (E_1(t-t_1)\ ^2 + E_2(t-t_2)\ ^2 + 2Re\langle E_1(t-t_1)E_2(t-t_2)^*\rangle \tag{2.6}$$

By denoting $E_1(t-t_1)\ ^2$ by $I_1$ and $E_2(t-t_2)\ ^2$ by $I_2$ and the time delay

$(t_1-t_2)$ between $E_1$ and $E_2$ by $\tau$ (since $E_1$ and $E_2$ are assumed to be stationary fields), Eq (2.6) can be rewritten as

$$IQ = I_1 + I_2 + 2\mathrm{Re}\langle E_1(t+\tau)E_2(t)^*\rangle \qquad (2.7)$$

The quantity $\langle E_1(t+\tau)E_2(t)^*\rangle$ is defined as the mutual coherence function $\Gamma_{12}(\tau)$ where the subscripts denote the points between which the coherence is measured.

Complex Degree of Coherence. The complex degree of coherence, $\gamma_{12}(\tau)$, is a normalized form of the mutual coherence function. It is defined as

$$\gamma_{12}(\tau) = \Gamma_{12}(\tau)/[\Gamma_{11}(0)\Gamma_{22}(0)]^{1/2} \qquad (2.8)$$

where $\Gamma_{11}(0) = E_1(t)E_1(t)^* = I_1$ and $\Gamma_{22}(0) = E_2(t)E_2(t)^* = I_2$. IQ can now be written as

$$IQ = I_1 + I_2 + 2[(I_1I_2)^{1/2}]\mathrm{Re}\,\gamma_{12}(\tau) \qquad (2.9)$$

Visibility. One way to conduct Young's two slit experiment is under what Zernike refers to as best conditions (1:10). These are $I_1 = I_2$ and the path differences are small. If $\gamma_{12}(\tau)$ is rewritten as a magnitude times a phase, i.e.

$$\gamma_{12}(\tau) = |\gamma_{12}(\tau)|\exp(i\phi_{12}(\tau)) \qquad (2.10)$$

where $\phi_{12}$ represents the difference in phase due to the path lengths, and $|...|$ indicate the magnitude of the quantity they enclose. Eq (2.9) may be rewritten as

$$IQ = 2I_1[1+|\gamma_{12}(\tau)|\cos\phi_{12}(\tau)] \qquad (2.11)$$

This results in a series of light and dark fringes appearing on plane B. The visibility of the fringes, V, is defined as (1:8)

$$V = (I_{max} - I_{min})/(I_{max} + I_{min}) \qquad (2.12)$$

Applying Eq (2.12) to Eq (2.11) yields

$$V = |\gamma_{12}(\tau)| \qquad (2.13)$$

The significance of this result is that the modulus of the degree of coherence, $|\gamma_{12}(\tau)|$, can be directly related to the measured visibility of the fringes (2:511). This result will prove useful later in this development.

Quasi-monochromatic Light Sources. When the light source emits quasi-monochromatic light, $\Gamma_{12}(\tau)$ is called the mutual intensity function and is denoted by $J_{12}$. The complex degree of coherence is still known as such but is now denoted by $\mu_{12}$. The source in the optical system being modeled is actually non quasi-monochromatic. A filtering operation (described in the section on calculating the mutual intensity function) takes place which effectively separates the incoming radiation into a series of quasi-monochromatic sources.

van Cittert-Zernike Theorem. Consider the situation depicted in Figure 2.3 below.

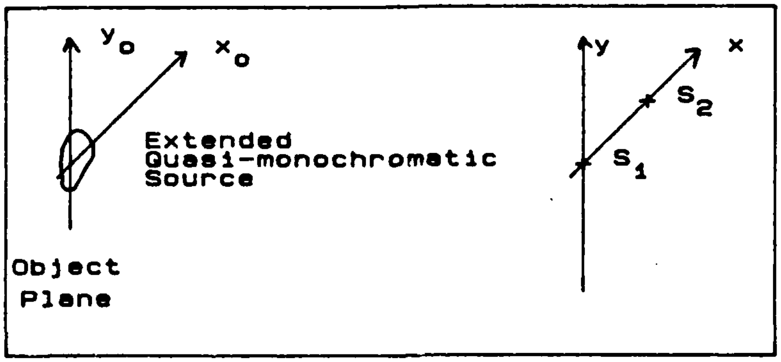



Fig 2.3. Example Situation.

The van Cittert-Zernike theorem, then, states that

> ...the complex degree of coherence, which describes the correlation of vibrations at a fixed point $S_2$ and a variable point $S_1$ (see Figure 2.3) in a plane illuminated by an extended quasi-monochromatic primary source, is equal to the normalized complex amplitude at the corresponding point $S_1$ in a certain diffraction pattern, centered on $S_2$. This pattern would be obtained on placing the source by a diffracting aperture of the same size and shape as the source, and on filling it with a spherical wave converging to $S_2$, the amplitude distribution over the wavefront in the aperture being proportional to the intensity distribution across the source (2:510).

Therefore, the source or object intensity distribution may be found by taking the inverse transform of the complex degree of coherence. This is the basic underlying principle on which the optical system to be modeled operates. One last useful observation is that the sources considered here are real and the Fourier transform of such a source is complex symmetric due to the hermitian nature of the transform (6:193). This means that if the complex degree of coherence $\mu_{12}$ is measured at a particular spatial frequency $f_{x1}$ then it is automatically known at the

symmetric spatial frequency $-f_{x1}$. It is simply the complex conjugate of the value of $\mu_{12}$ at $f_1$.

Linking of Theory to the Optical System

The optical system to be modeled behaves somewhat like a coherent imaging system. According to Goodman (7:106-110), if the object is illuminated by coherent light the impulse responses comprising the image must be added on a complex amplitude basis. Therefore, a coherent imaging system is a linear system with respect to complex amplitude 7:107). In a coherent imaging system, the image is the convolution of the image predicted by geometrical optics with an impulse response determined by the exit pupil of the system (7:105). This is denoted by

$$u_i(x_i,y_i) = \iint_{-\infty}^{\infty} h(x_i-x_o,y_i-y_o)u_o(x_o,y_o)dx_o dy_o \qquad (2.14)$$

where $u_i$ and $u_o$ are the image and object amplitude distributions and

$$h(x_i,y_i) = \iint_{-\infty}^{\infty} P(\lambda d_i x,\lambda \phi_i y)\exp(-j2\pi(x_i x + y_i y))dxdy \quad (2.15)$$

and where P is the pupil function and $d_i$ the image distance (7:105). Eq (2.15) is in the form of a Fourier transform so that h is the Fourier transform of the pupil function P.

Applying the convolution theorem of Fourier transforms to Eq (2.14) yields

$$G_i(f_x,f_y) = H(f_x,f_y)G_o(f_x,f_y) \qquad (2.16)$$

where (denoting a Fourier transform by $\mathcal{F}$)

$$G_i(f_x,f_y) = \mathcal{F}(u_i(x_i,y_i))$$
$$G_o(f_x,f_y) = \mathcal{F}(u_o(x_o,y_o))$$
$$H(f_x,f_y) = \mathcal{F}(h(x,y))$$

Eq (2.16) describes a linear system. Since h(x,y) is the Fourier transform of $P(\lambda d_i x, -\lambda d_i y)$, the value of $H(f_x, f_y)$ is $P(-\lambda d_i x, -\lambda d_i y)$ or, if one assumes a reflected coordinate system, $H(f_x, f_y) = P(\lambda d_i x, \lambda d_i y)$ (7:110-111). The nature of H is that it allows all of the light at the sampled frequencies to pass through and completely attenuates the light at all other points. Inverse transforming $G_i$ yields the image $u_i$.

The input or object intensity distribution for the optical system being modeled is the mutual intensity function. H consists of points sampled by the optical elements of the system. $G_i$ is then a spatially filtered version of $G_o$. Inverse transforming $G_i$ will then produce the image. H determines the amount of spatial filtering. Consider this one-dimensional case. If $G_o$ was a rect($f_x$) and H was a rect($2f_x$), $G_i$ would be a rect($2f_x$). Spatial frequencies greater that 0.5 are filtered out. This obviously degrades the quality of the final image. This is why the effect of the aperture or pupil function is of such interest.

Calculation of the Mutual Intensity Function. The scene of interest is considered to be spatially incoherent, temporally stationary, and non quasi-monochromatic. The source may therefore be thought of as a collection of m independent oscillators operating at their own individual frequencies and radiating a complex field $E_m(t)$. The total field present at each of lenses $L_1$ and $L_2$ can then be thought of as the sum of the fields due to each oscillator. This is denoted by

$$E_1(t) = \sum_m E_{m1}(t) \qquad (2.17)$$

$$E_2(t) = \sum_m E_{m2}(t) \qquad (2.18)$$

A linear filtering operation now takes place in order to separate the incoming fields for detection by the array of n linear detectors.

The filters are assumed to be narrow band (to fulfill the requirement of quasi-monochromatic light), are identical for each lens, and have a Fourier transform $G_n$. The received signal is then $Sf_n(t) = E_n(t) * g_n(f_n)$ (n = 1,2,...) (n indicates which detector is being analyzed). The mutual intensity function can then be rewritten as just

$$\Gamma_{12} = \langle Sf_{n1}(t)Sf_{n2}(t)^{*}\rangle = Sf_{n1}(t)Sf_{n2}(t)^{*} \qquad (2.19)$$

This illustrates that the entire continuous mutual intensity function is not calculated but instead only at the n specific frequencies. A sampled version is obtained instead. This results in some degradation of the image. The effects of the number of frequencies sampled on image quality is therefore one of the goals of this thesis.

<u>The Aperture Function</u>. The linear detector array cannot (unfortunately) contain enough detectors to detect every frequency present. Therefore, assume that the detector array contains four detectors at frequencies $f_1$, $f_2$, $f_3$, $f_4$ for the following example. Figure 2.1 illustrated the system geometry. The mutual intensity function is measured at intervals along $\Delta\Theta$ as the system passes across the scene. The magnitude of the four frequencies measured by the detector array is recorded for each interval. A polar plot of the frequency versus $\Delta\Theta$ is shown in Figure 2.4.

The system is able to obtain resolution in the dimensions of slant range and azimuth. The are denoted by $f_r$ and $f_a$ in Fourier transforms as in Figure 2.4. The directions of the radio lines can be thought of two ways the direction is mathematically the result of the difference of two unit vectors denoting the positions of the lens relatively to the

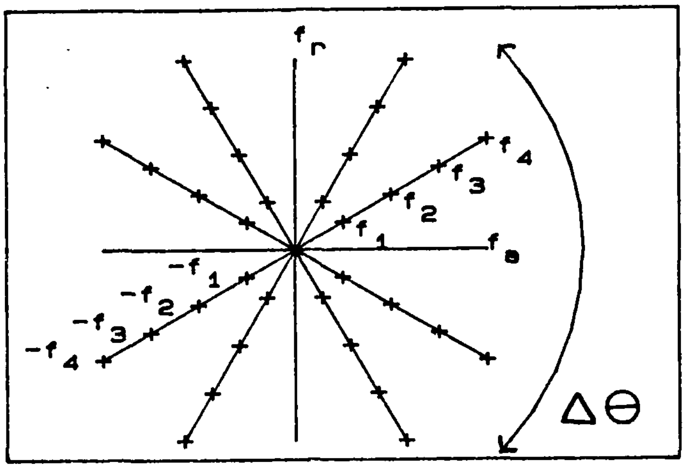



Fig 2.4. Polar Plot of Frequencies Sampled.

source (see Appendix A). The direction can be thought of conceptually as the direction of the lens separation relative to the source as the lens move.

The resolution along the two dimensions is determined by $\Delta\theta$ the frequency spacing. This is best illustrated by the following special cases. When the lenses are (hypothetically) infinitely far away from the source such that they lie in the plane of the source, the sampled frequencies all lie along $f_r$ yielding resolution in slant range but none in azimuth. When the lenses are in a broadside position or directly overhead, no resolution along $f_r$ is possible because all the sampled frequencies lie along $f_a$. The resolution along $f_a$ is determined by the spacing between samples.

It is clear from Figure 2.4 that not all frequencies are sampled. The hermitian nature of the mutual intensity function described earlier allows the determination of the function $\mu_{12}$ at the points $-f_1$, $-f_2$, $-f_3$, $-f_4$ because their value is equal to the complex conjugate of $\mu_{12}$ at the points $f_1$, $f_2$, $f_3$, $f_4$. However, the amount of frequency coverage is

limited by $\Delta\theta$ and the number of detectors. The extent of $\Delta\theta$ determines the overall shape of the aperture function and the frequencies at which detectors are present determine which frequencies are sampled and which are not.

Final Output of the Optical System. The final output is an image of the original source obtained by inverse Fourier transforming a sampled version of the mutual intensity function as governed by Eq (2.16). This results in the degradation of the image since not all of the frequency components of the mutual intensity function are present. This thesis determined what effect various aperture functions had on the final image.

# III. THE COMPUTER MODEL

This chapter discusses the development and operation of the computer model of the optical system. The supporting hardware and software will be described. A brief explanation of the overall flow of information will be given. Appendix B contains the program listings and operating instructions. A printout of a sample run is also provided.

## Computer Support

The model was developed at the Electro-Optics Branch of the Air Force Wright Aeronautical Laboratory. The computer employed was a Hewlett-Packard (HP) 9000 which ran under the UNIX operating system. The graphics support consisted of an HP 2623a graphics terminal with an internal printer and an HP 7550a plotter. The computer language used to construct the model was Fortran 9000 (a version of Fortran 77). The HP 9000 had several important qualities which are enumerated below.

One of the best features was that of being a virtual memory machine. This eliminated program size considerations. The only concern was speed of execution since the machine can accomodate a program of any size. This was one less problem to have to consider and therefore allowed more concentration on the physics of the thesis. The two megabytes of memory were more than sufficient for the model.

The graphics package (named Advanced Graphics Package or AGP) was also easy to use (if not a little confusing to learn). The only bad feature was that there was no way of hiding lines if the programmer did not know what kind of data was coming. This is why the plots that appear later in this thesis are only one quarter of the front half of the picture. This eliminated many of the confusing lines and was possible due to the symmetry of the information.

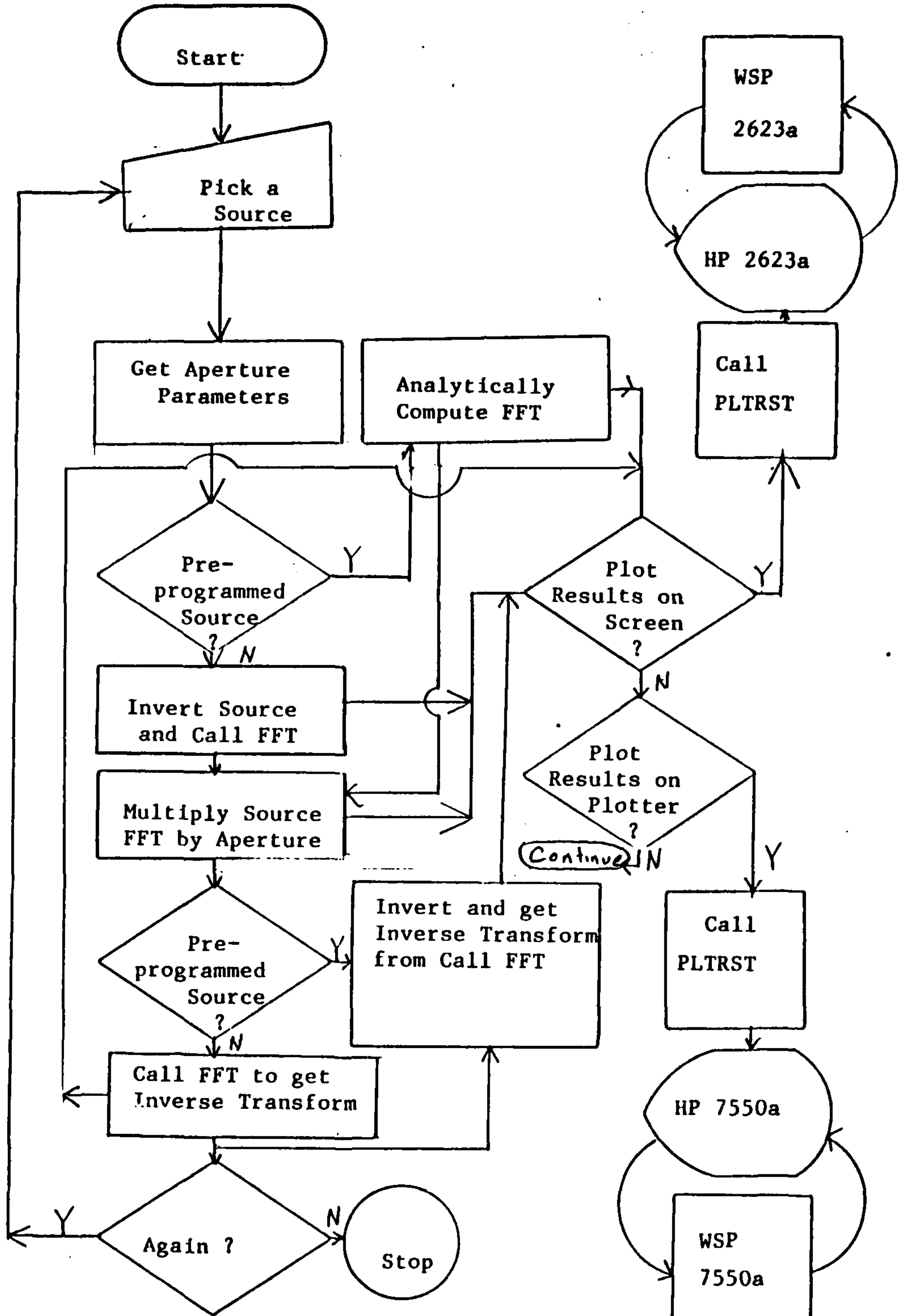



Fig 3.1. Flowchart of Computer Model

The feature that made AGP so useful was the concept of a Work Station Program (WSP). Each graphics device has an associated WSP which takes care of all device dependent affairs allowing the programmer to use the same plotting programs on different devices. The only changes occur in device initialization and in the calling sequences. Appendix B contains more details on this.

<u>The Computer Program</u>

The preceding flowchart illustrates the various parts of the computer model and how they interact. The model is invoked by typing "synapt" (SYNthetic APerTure) on the terminal. This is also the name of the main program. See Appendix B for more detailed running instructions.

The model first initializes all graphics devices. It is then ready to find out what type of source is to be imaged. The model contains five preprogrammed sources and will allow the user to put in his own. The five preprogrammed sources are a point source, a two-point source, an edge, a slit, and a circle. The model analytically computes the Fourier transform of the first four sources and invokes a Fast Fourier Transform (FFT) subroutine to compute the Fourier transform of the circle and that of the user's own.

The model then asks for information regarding the aperture or pupil function. The first variables required are the range to the source and the lens separation. These variables determine what frequencies will be sampled (see appendix A for a rigorous derivation of this). A maximum $\Delta\Theta$ is also calculated from the range information, collector speed, and

collector stability information provided by the user. The user then is asked for $\Delta\Theta$. An upper and lower limit are required. This $\Delta\Theta$ is compared to the maximum possible $\Delta\Theta$ calculated above. If the calculated $\Delta\Theta$ is exceeded, the user must start over with new range and lens separation information.

It was decided to use rectangular symmetry instead of radial symmetry in the aperture and source distribution because an FFT subroutine that worked with radial symmetry could not be found. The points in the pupil function were modelled as having a radial distance ( $(f_r^2 + f_a^2)^{1/2}$ where $f_r$ and $f_a$ are the rectangular components of the frequency in range and azimuth respectively) which was the equivalent to the radial frequency at that point. Each element of the pupil function either transmits entirely or attenuates entirely. A picture of the pupil function appears in Figure 3.2.

All of these frequencies are not sampled. The frequencies that are sampled are

$$f = 2(f_m/c) \sin(\alpha/2) \qquad (3.1)$$

where $f_m$ is the frequency of a filter and $\alpha$ is the angle between the lenses formed by $R_1$ and $R_2$ (see appendix A for supporting information and a derivation of Eq 3.3). The spatial frequencies f that are sampled in the optical system being modelled are 0 to 64 1/m based on ranges of 1 to 3 kilometers, a lens separation of 0.5 meters, a wavelength $(c/f_m)$ band of 8 to 12 μmeters, and a maximum pupil radius of 16 bits in a 256 by 256 array of points that modelled the pupil. The model computes the

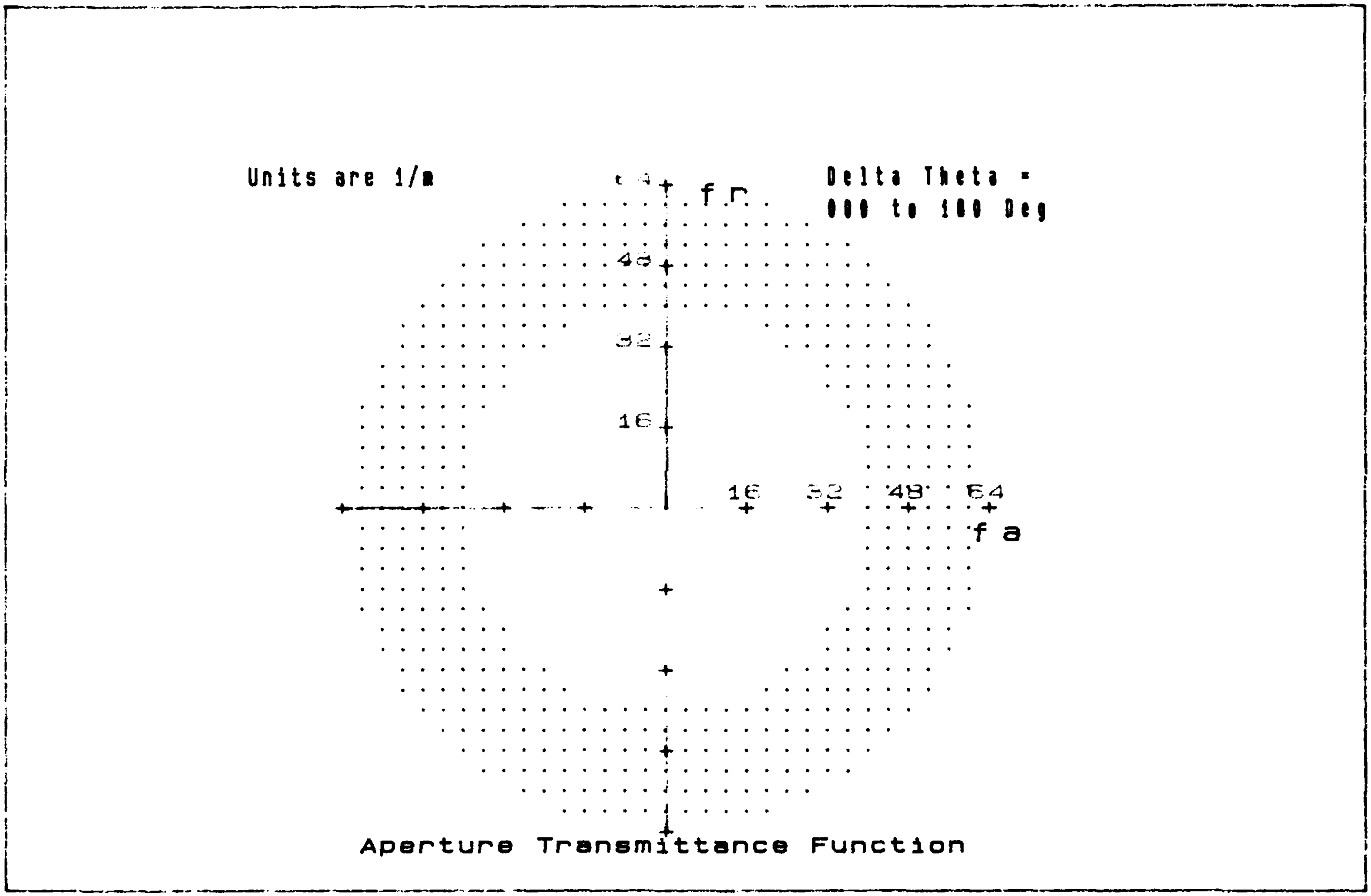



Fig 3.2. Example Plot of Aperture or Pupil Function

upper and lower frequencies based on the information indicated above and shades the appropriate area of the pupil to reflect the operating conditions.

The source FFT now undergoes an inverting process. The FFT subroutine normally places the high frequency components in the middle of the transformed array and the low frequency components at the corners. This process allows the FFT to appear in its more commonly recognized form with the low frequencies in the center and the high frequencies on the outside edges. See Appendix B to see how it is implemented.

The source FFT now is multiplied by the pupil function. The numbers to be multiplied in general are complex and the operation is of the form

$$( a + ib ) ( c + id ) = \text{Answer} \tag{3.2}$$

where i is the square root of negative one, a + ib represents the value of a point of the FFT of the source, and c + id is the value of a point in the pupil. However, since the components of the pupil function are entirely real ( d = 0 ), the result is

$$ac + icb = \text{Answer} \tag{3.3}$$

This result is stored in the array that originally held the source FFT.

The final answer is now obtained by taking the inverse Fourier transform (IFT) of the result of Eq 3.3. The data is inverted as before in order for the answer to appear in its original form. The user can get a picture of this and compare it to the original source to see just how well his system has performed.

There are two subroutines other than the main program which do a majority of the work. The most important is the

FFT subroutine. Figure 3.3 compares the results of an analytically computed 32 sinc (32 $f_x$) to the Fourier transform of a pulse with a width of 32.

The subroutine computes the transform by finding the terms of a Fourier series of the same function as if it were indeed periodic. A period of 256 units was empirically found to yield an acceptable accuracy as illustrated in Figure 3.3. Although a larger period was more desirable, lengthening the array also resulted in greatly increasing the time required by the computer to compute the FFT. The subroutine required the input of both a real and imaginary component. This resulted in requiring two 256 by 256 element arrays to adequately describe the source in two dimensions. A listing of the subroutine may be found in Appendix B.

The other subroutine is called PLTRST for PLoTReSulT. This subroutine carries out the graphics operations. The subroutine requires that the calling program indicate which plot is needed. The subroutine then draws the appropriate axes in the appropriate projection and types all of the appropriate headings, titles, and other markings. This is output to the appropriate device as the user has indicated. The WSP takes care of the actual drawing. See Appendix B for more details.

Several other subroutines are also invoked in the model. However, they perform only support functions and it is therefore unnecessary to go into detail here on what they do. Again, Appendix B contains more information on this matter.

Outputs

The outputs of the model are all graphical in nature. The model will plot on both the terminal and the plotter. There are a total of

five possible outputs on any one run of the model. These are (1) of the source irradiance distribution, (2) the aperture transmittance function, (3) the FFT of the source, (4) the product of the source and the aperture, and (5) the inverse FFT of the product of the source and the aperture or the image.

The scales in the object and the image plane are in terms of a dimensionless variable V defined as

$$V = (2\pi a/(\lambda d_i))x_i \qquad (3.4)$$

where a is the pupil radius, $\lambda$ is the cutoff wavelength, $d_i$ is the image distance, and $x_i$ is the position of a point in the image. The size of a is 0.5 m which was found from the relationship found in reference 7:112 for the cutoff frequency of a coherent transfer function

$$f_c = 1/(2\lambda d_i) \qquad (3.5)$$

where l is the diameter of the pupil. Equating this to Eq 3.1 resulted in l = d where d is the lens separation. The value of $d_i$ is fixed due to the fixed focal length of the lenses. The value of $\lambda$ is taken to be 8um at the radius of the pupil. A sample object size of 3.125 cm was used to arrive at a corresponding value of V of $2000\pi/512$. This object was assumed to be 32 bits wide in the 256 by 256 array representing the source. As noted earlier, only one quarter of the front half of the plot is illustrated. Therefore, the value of V must be doubled to find the total width or length of the object. The tick marks in the plots of the object and the image plane represent 8 bits in the source array. This is why the scale appears as is illustrated in Fig 3.4.

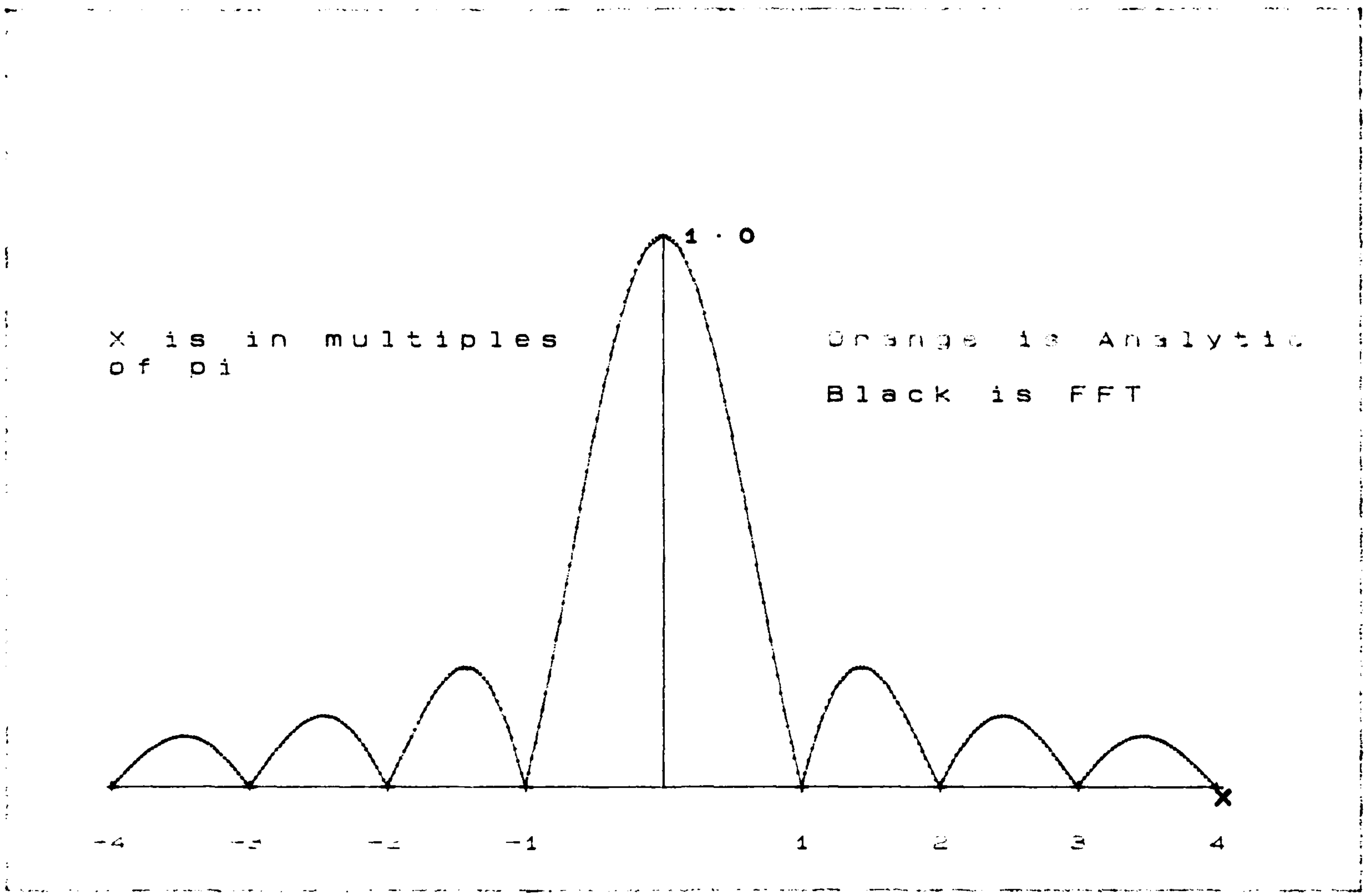



Fig 3.3. Comparison of Analytic Sinc and FFT of Pulse

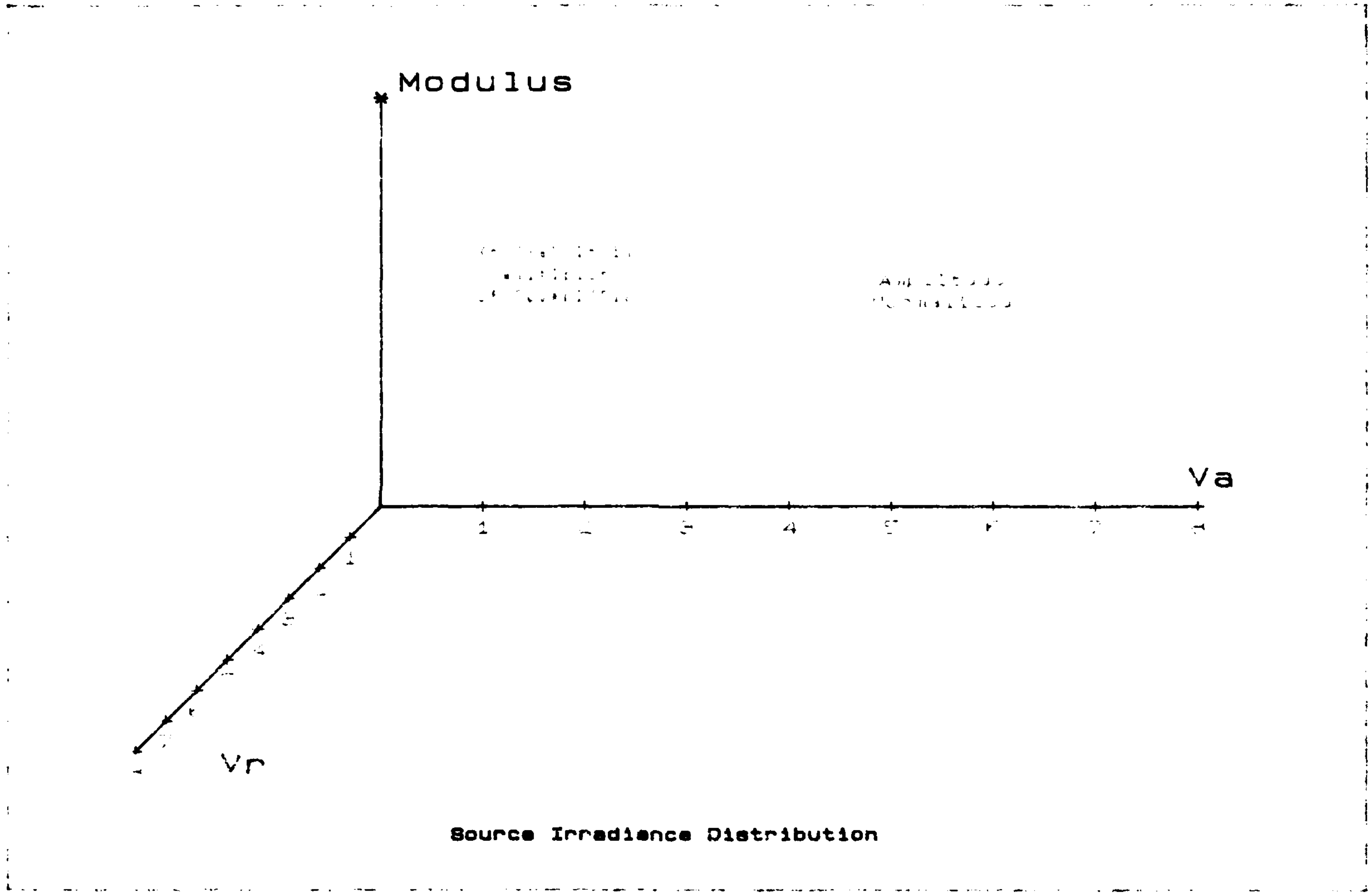



Fig 3.4. Example Plot of Point Source Distribution

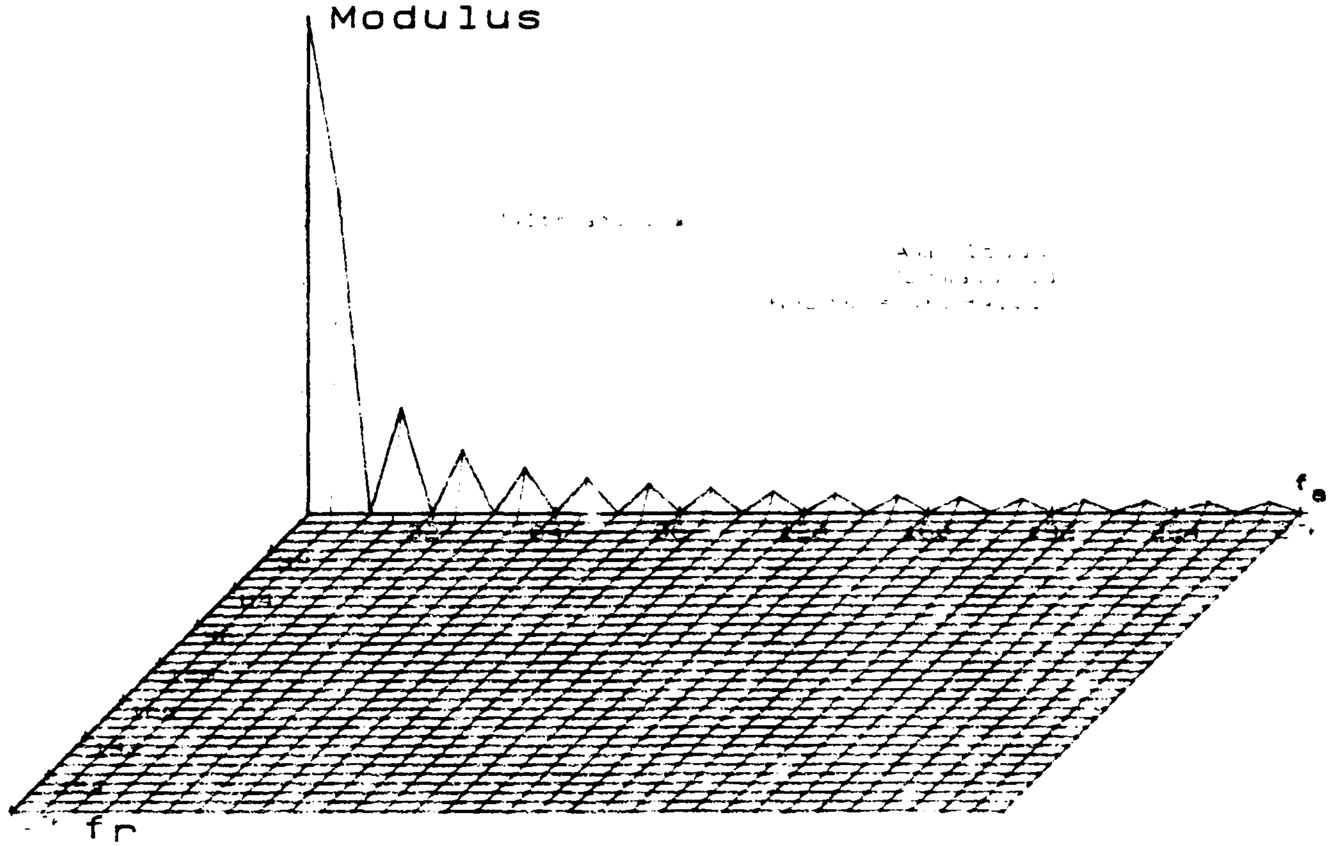



Fig 3.5. Example Plot of FFT of 6.25 cm Edge

The other outputs occur in the frequency plane. The scale is as described above in the pupil function. The axes are labelled in range ($f_r$) and azimuth ($f_a$). A sample of the modulus versus the spatial frequencies of the FFT of a 6.25 cm wide edge appears in Fig 3.5.

All of the plots are normalized in amplitude. In all cases, the modulus of the amplitude is plotted. The factor by which the information is normalized is shown on the plots as Rnorm. The preprogrammed sources are assumed to have an amplitude of one originally.

# IV. RESULTS

This chapter contains the results of successive runs of the computer model. Five types of sources were imaged through five types of apertures. The results are in the form of plots of the image of each of the five sources due to each aperture configuration. Each aperture and the images of the five sources due to each aperture appear on foldouts at the end of the chapter. R. Barakat (16:205-223) has also examined the effects of different apertures on the images of various objects. His results agree quite closely with the results obtained with the computer model.

The initial aperture reflected the best case scenario; i.e. all frequencies were allowed to pass below the cutoff wavelength of 8 μm. The next two apertures reflect how the images degrade as fewer and fewer of the low frequencies were allowed to pass. The images began to show the characteristics of edge enhancement (10:61-62) as the lower frequencies were filtered out. These apertures established a baseline from which apertures obtained under realistic conditions could be compared.

Figures 4.1 and 4.2 illustrate the process of edge enhancement for the case of an edge source with a uniform amplitude distribution and a width of 6.25 cm. The X's in Figure 4.1 indicate the bounds on the spatial frequencies of the Fourier transform of the edge which are allowed to pass by the actual system. This figure reflects a lens separation of 0.5 m and a range of 1 km. These two factors plus the 8 to 12 μm bound on the detectable wavelengths combine to yield a limit on the spatial frequencies of approximately 40 to 64 cycles per meter.

Figure 4.2 is the image obtained by analytical methods with the above mentioned system parameters. The edge falls at $X_1$ equal 4 in Figure 4.2. This confirms that edge enhancement takes place.

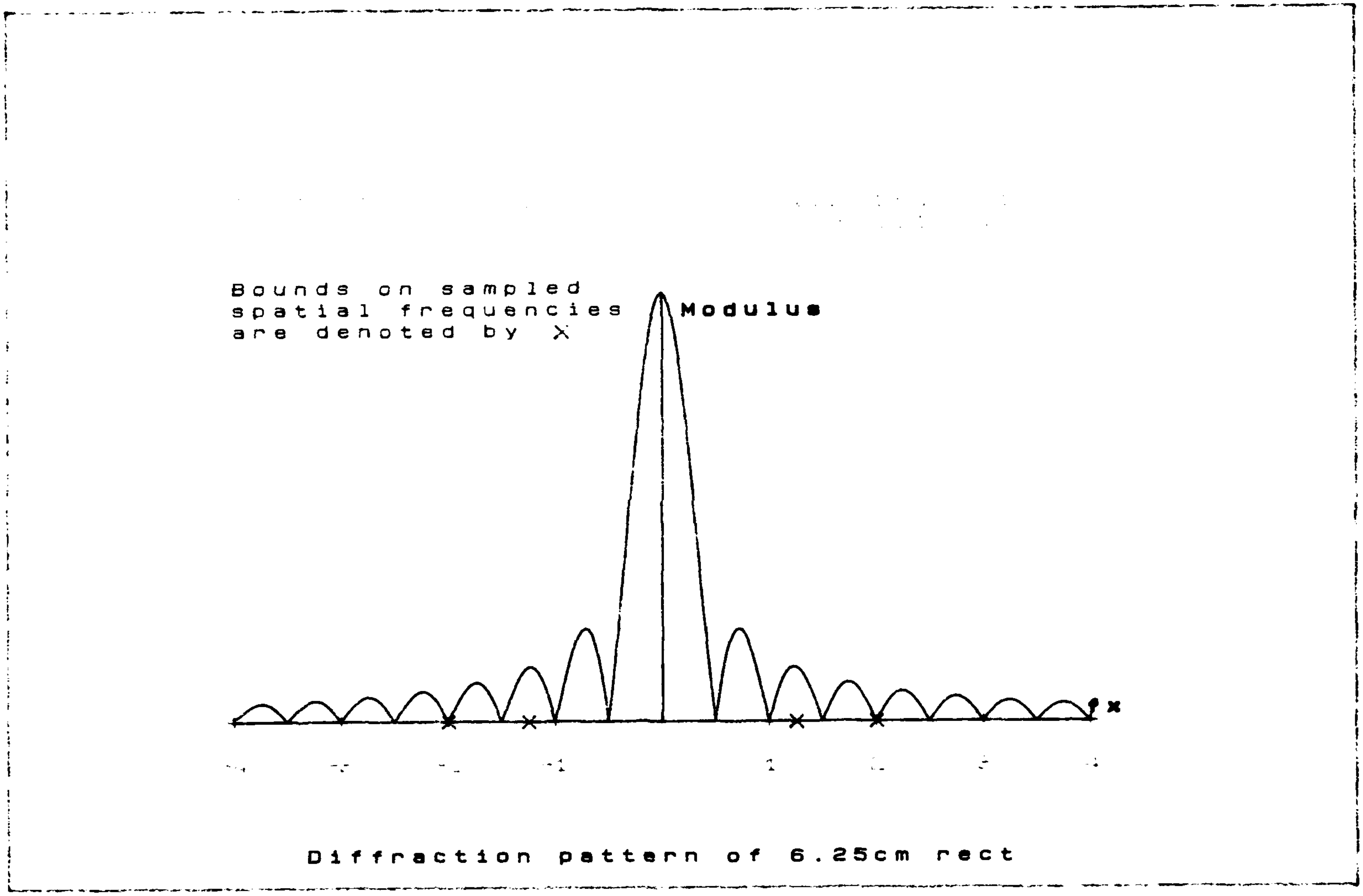



Fig 4.1. Bounds on Spatial Frequencies of Analytic Image

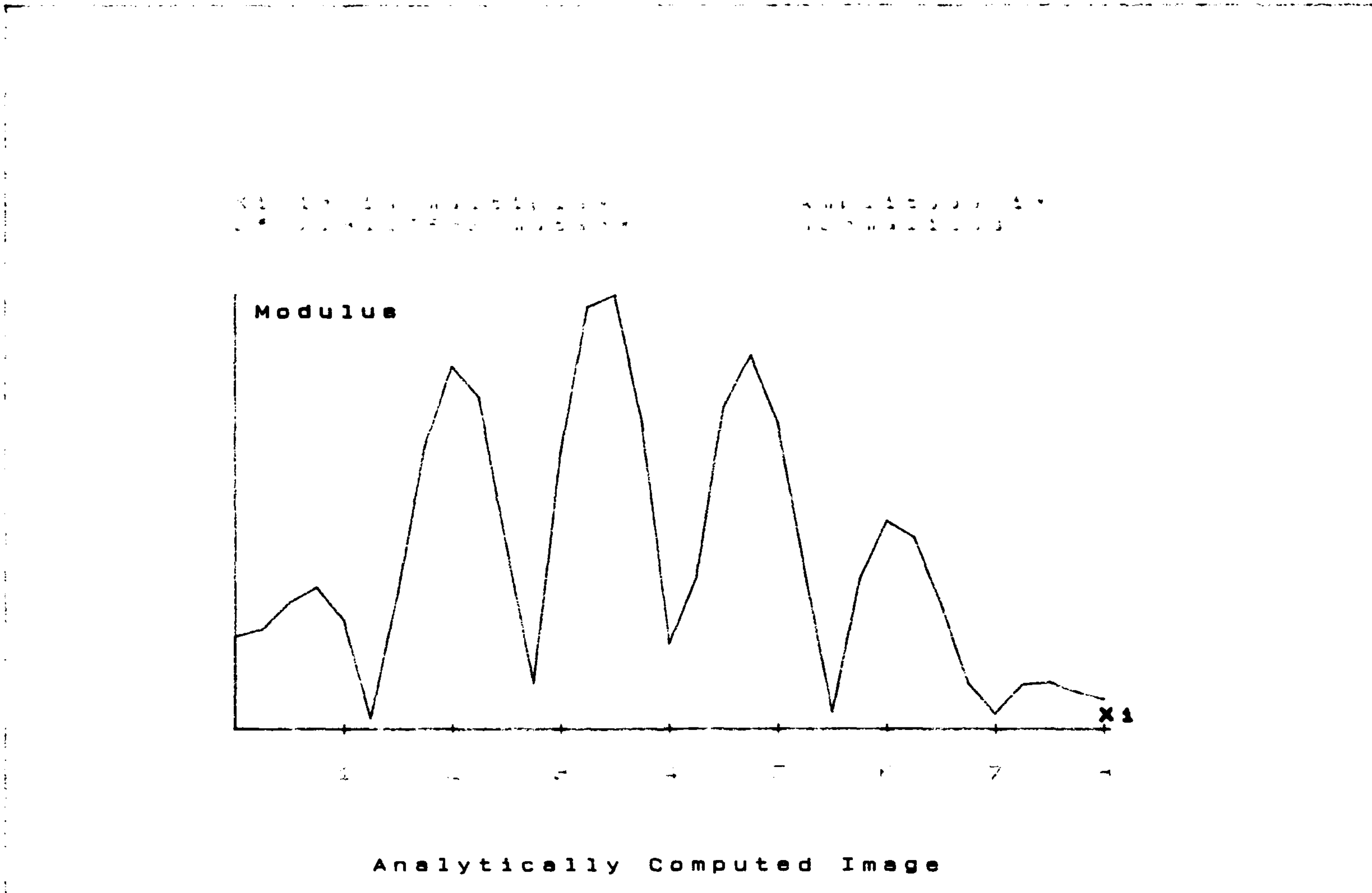



Fig 4.2. Image of 6.25cm Wide Rect

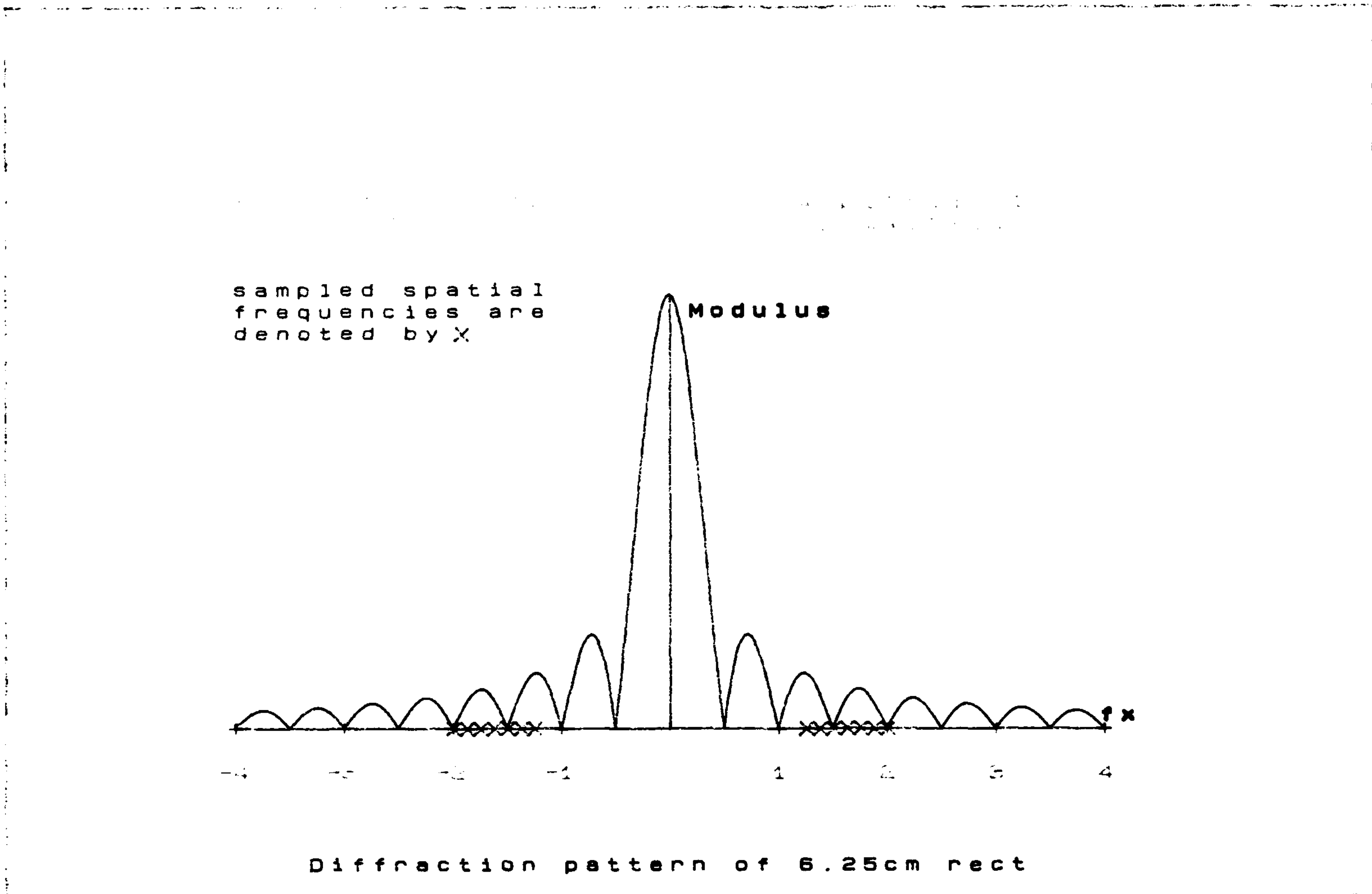



Fig 4.3. Sampled Spatial Frequencies in System Image

Figures 4.3 and 4.4 compare how the computer model behaves under identical operating conditions. Figure 4.3 illustrates which frequencies are sampled by the computer model. The analytical case above assumed all spatial frequencies between 40 and 64 cycles per meter were sampled. Figure 4.3 shows that this is not the case in the computer model. The bounds on the spatial frequencies remained the same but samples were taken only at 4 cycles per meter intervals as indicated by the X's in Figure 4.3. Figure 4.4 shows the image obtained by the passive system through the computer model when only discrete samples of the Fourier transform are taken. Figure 4.4 agrees very closely with Figure 4.2 which would be the best image that could be obtained. This comparison was the final test in validating the computer model.

The five sources imaged were: a point source, two point sources with a separation in $V_r$ of 4, an edge (that was modeled by a rect with a width of 4 along $V_a$), a slit with dimensions of ($V_r$ X $V_a$) 4 X 2, and a circle with a radius of 4. The amplitude distribution was unity at all points on the sources. Figures 4.5 and 4.6 illustrate these objects. The Fourier transforms of these sources were taken as described in Chapter III and passed through five different apertures. The resultant images of these sources through each of the apertures appear on five foldouts at the end of this chapter along with the aperture used. The five cases are described below.

## Case I

The first case considered what images could be expected from a full aperture. The highest frequency present was 64 cycles per meter. The images are as expected with some ringing present. The impulse response

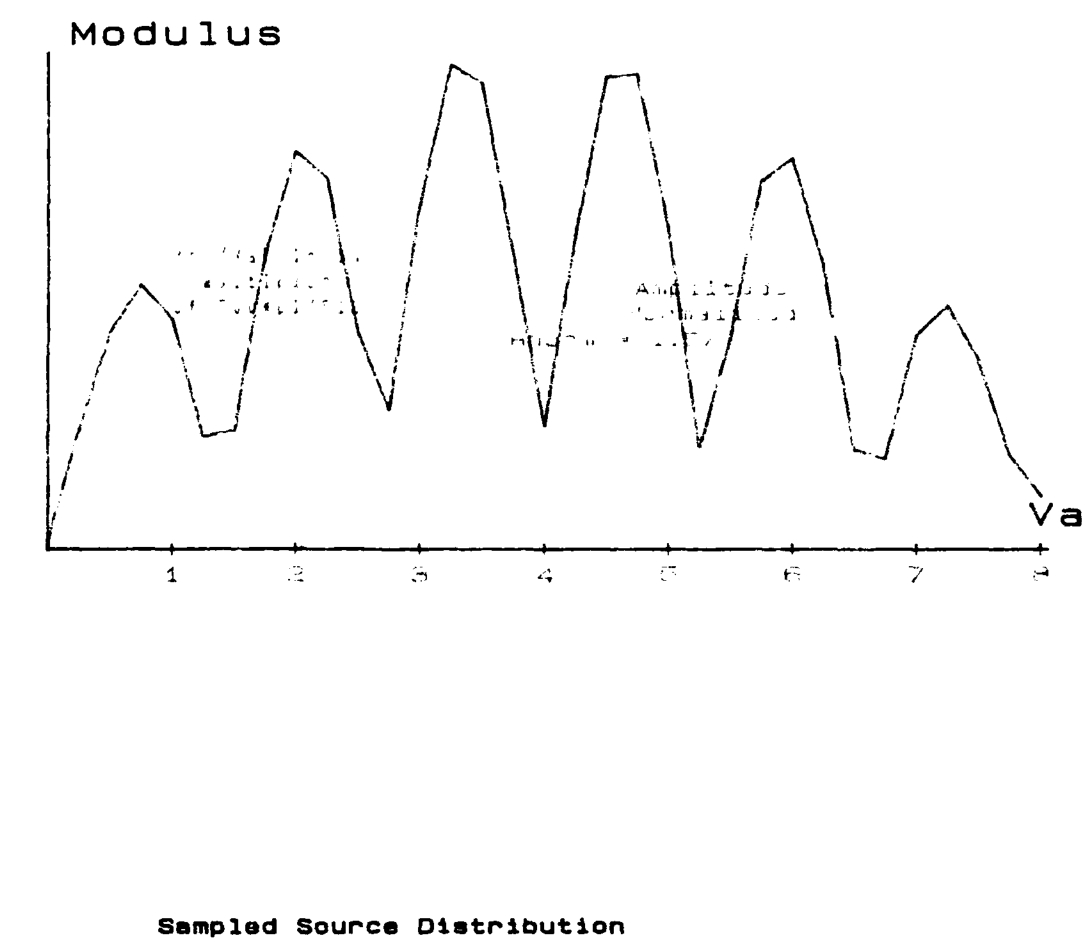



Fig 4.4. Image Obtained from Passive System

is good with low sidelobes. The images can be interpreted easily at this point. This case is reflected on the first foldout (Figure 4.7).

Case II

The second case used an aperture that passed only spatial frequencies between 16 and 64 cycles per meter. The impulse response now is more spread out with higher sidelobes. The images are no longer easily interpreted. Substantial ringing is beginning to occur. This case is illustrated on the second foldout (Figure 4.8).

Case III

This case reflects what happens to the images when only the spatial frequencies between 40 and 64 cycles per meter are sampled. This case reflects realistic spatial frequencies since they reflect realistic system parameters of 0.5 m for lens separation and 1 km for range. $\Delta\Theta$ is considered to be 180 degrees in this case still. The images have degraded even more with more ringing present along $V_a$. This is reflected in the third foldout (Figure 4.9).

Case IV

This case reflects the system performance under the conditions in Case III with the addition of a realistic $\Delta\Theta$. This case is a reflection of the kind of data that can be expected from a passive synthetic aperture system. The velocity and stability used in calculating $\Delta\Theta$ were 880 ft/s and 10 s. This resulted in a $\Delta\Theta$ of approximately 106 degrees which was centered on both sides of $f_a$. The impulse response has even higher sidelobes and the images have become slightly noisier than in the previous case. The data for this case is shown in the fourth foldout (Figure 4.10).

Case V

The last case was intended to show what happens under a slightly different system configuration. The lens separation has been halved to 0.25 m. The sampled spatial frequency range is now approximately 20 to 32 cycles per meter. The impulse response now exhibits wider and more spread out sidelobes along Va than in previous cases. This is also reflected in the images of the other sources. This is illustrated in the last handout (Figure 4.11).

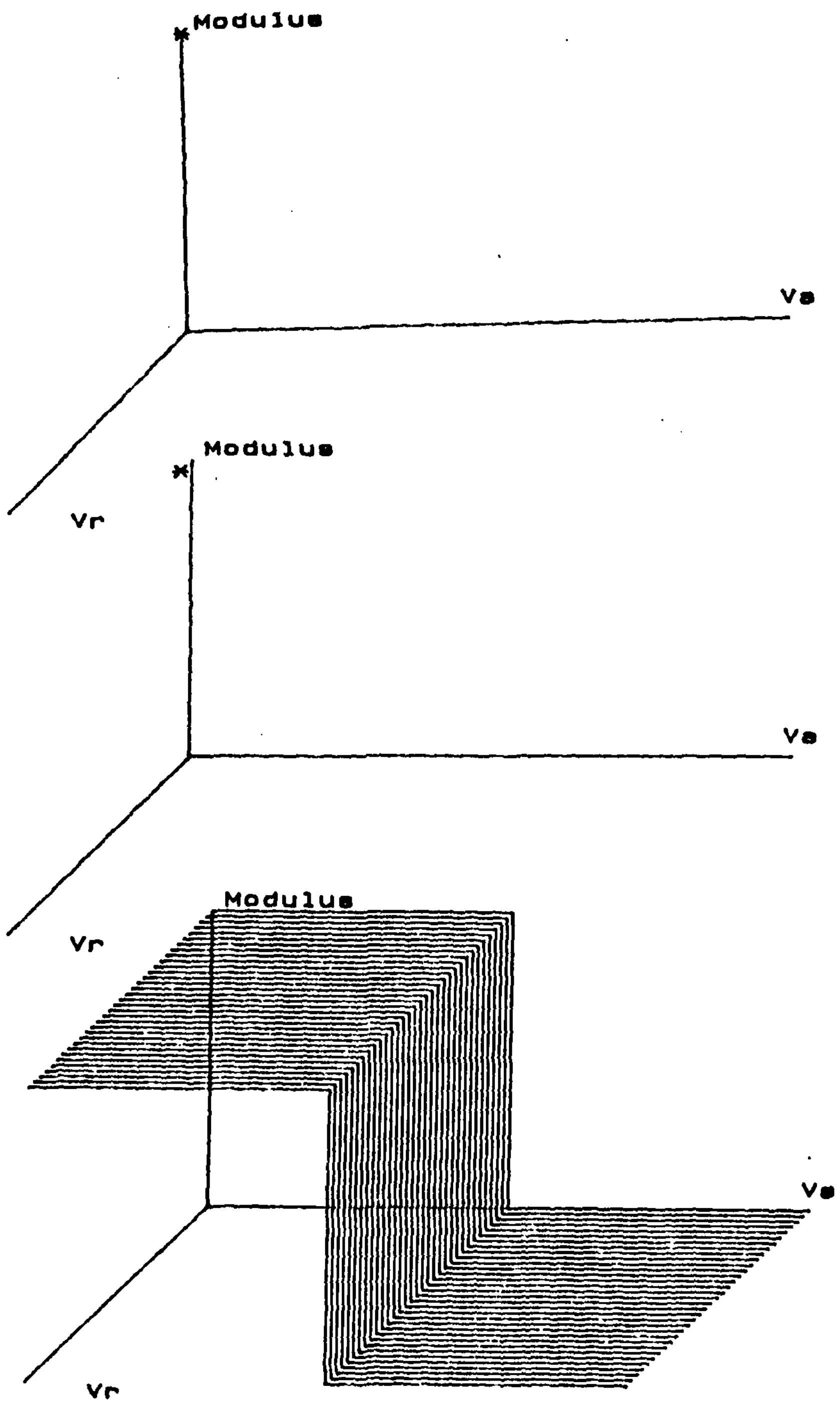
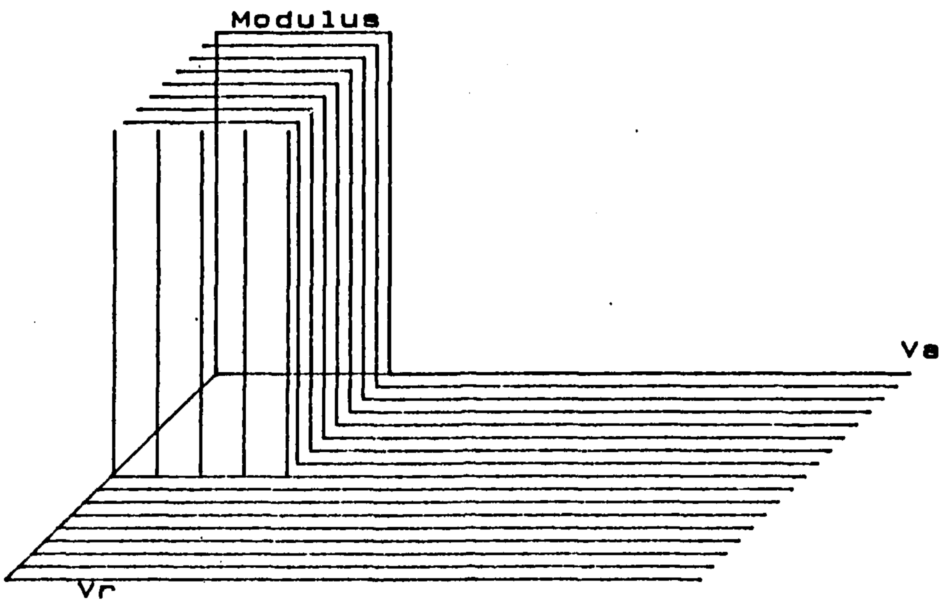



Fig 4.5. One Point (top), Two Point (middle), and Edge Objects



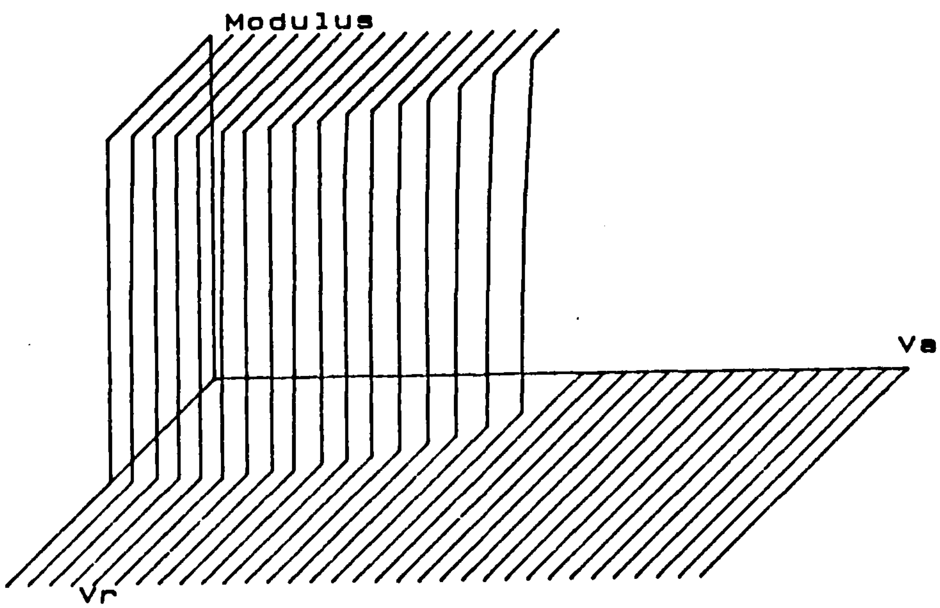




Fig 4.6. Slit (top) and Circle Objects

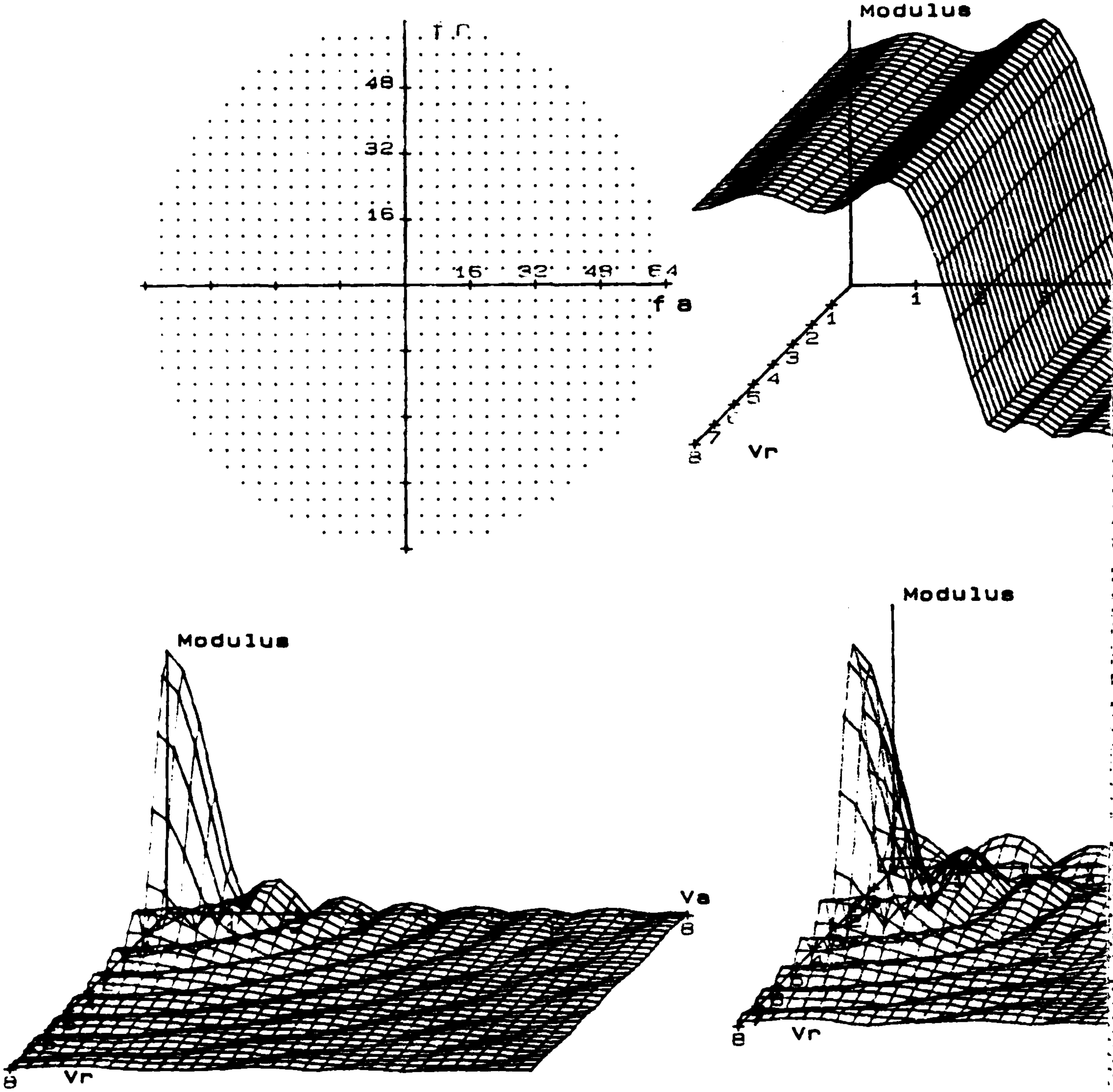
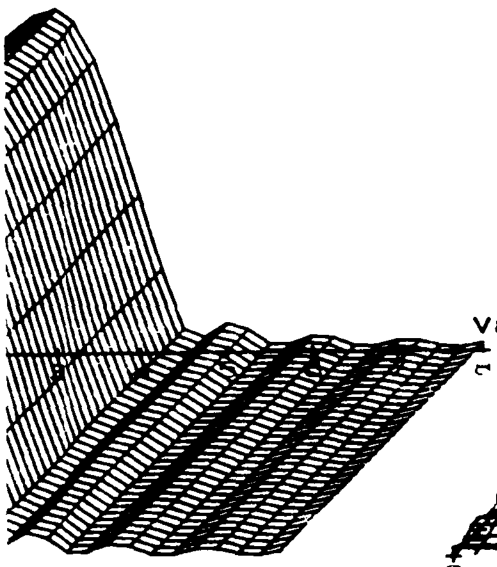
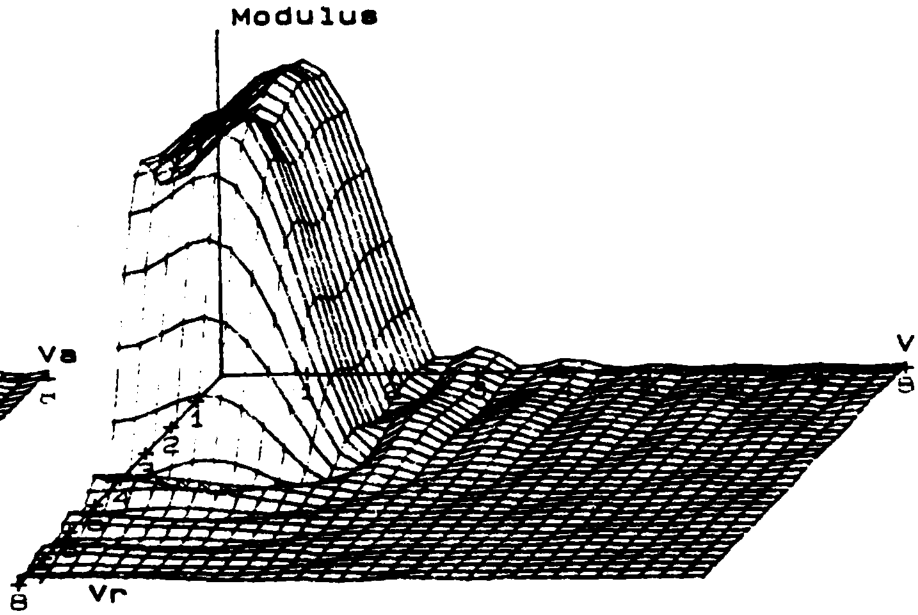
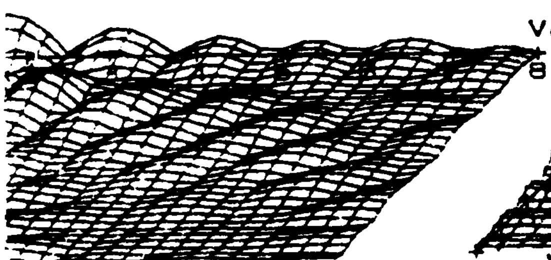
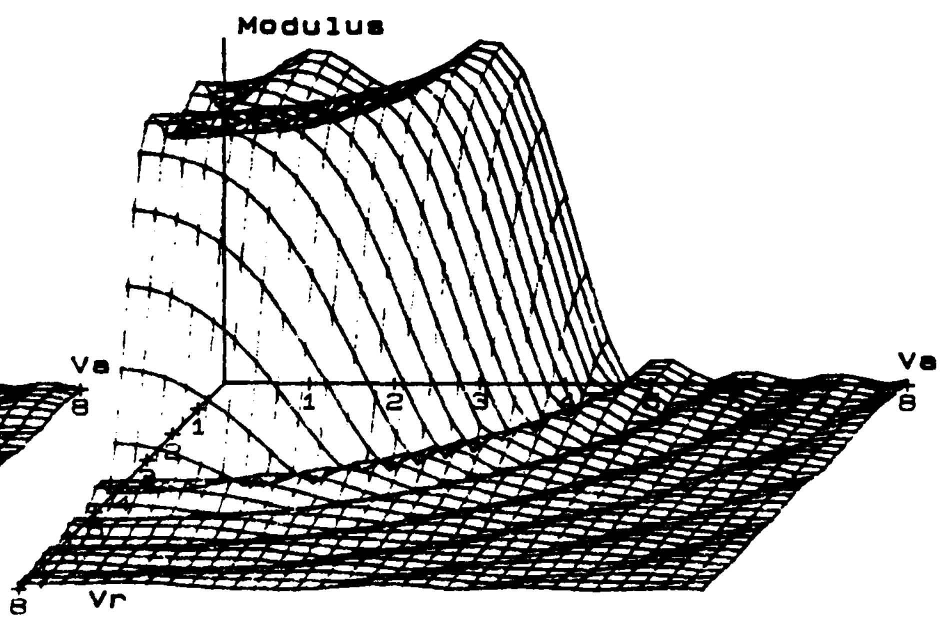



Fig 4.7. Case I Results


Va


Modulus
Va
8
1
2
3
4
8
Vr


Va
8


Modulus
Va
1
2
3
8
1
2
3
8
Vr

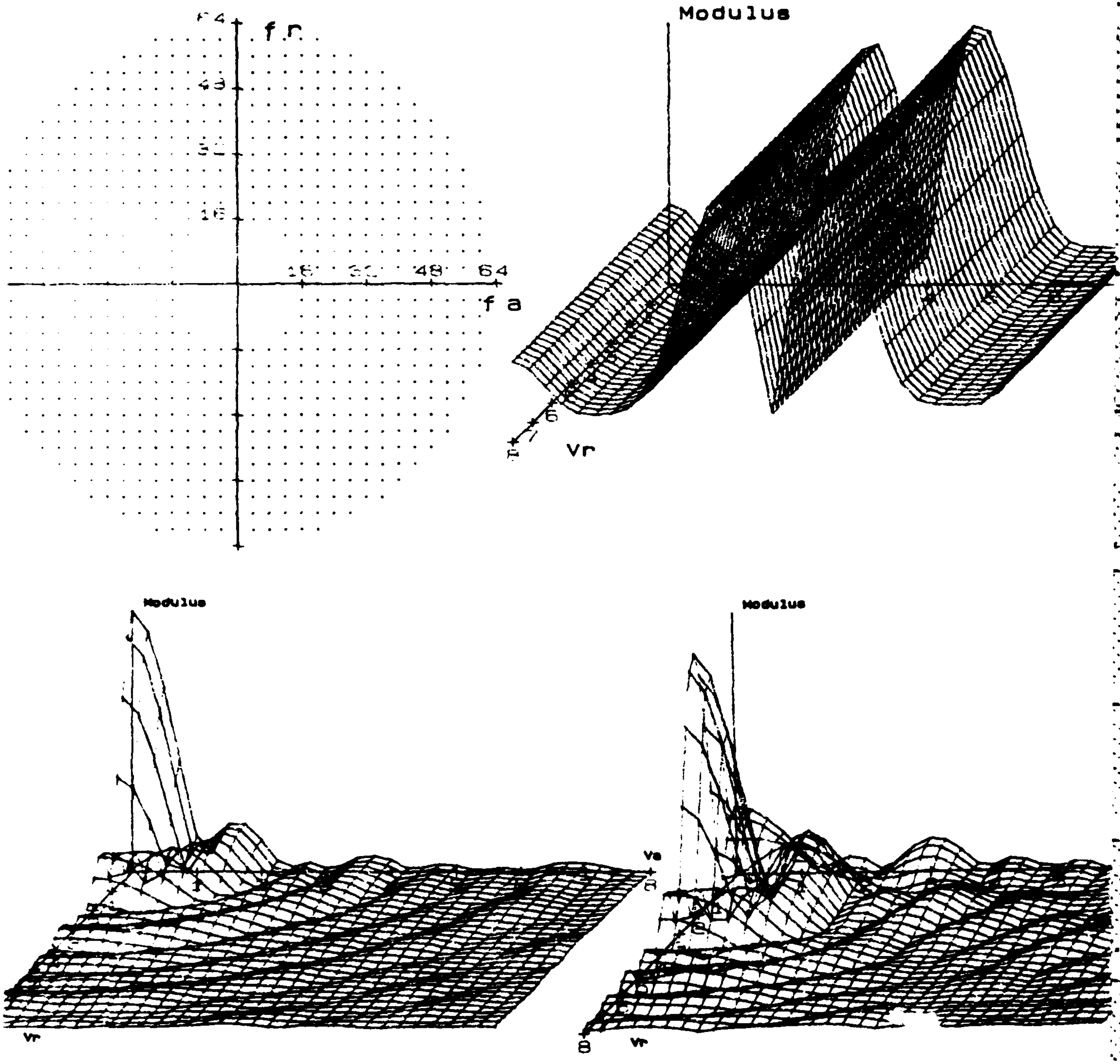
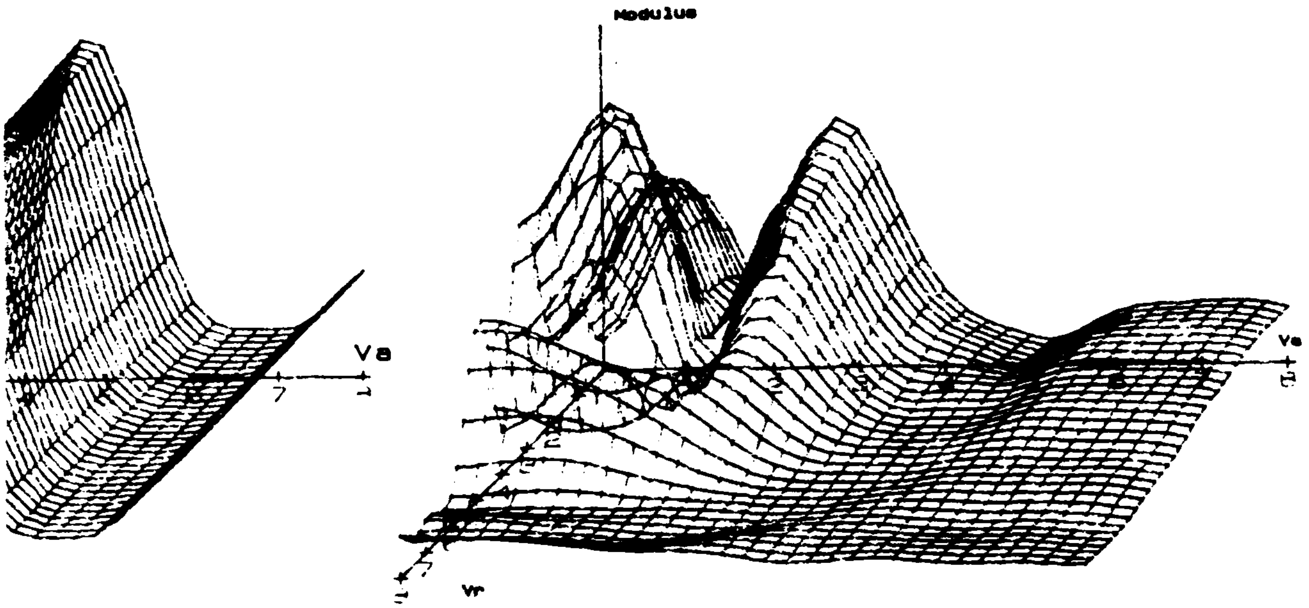
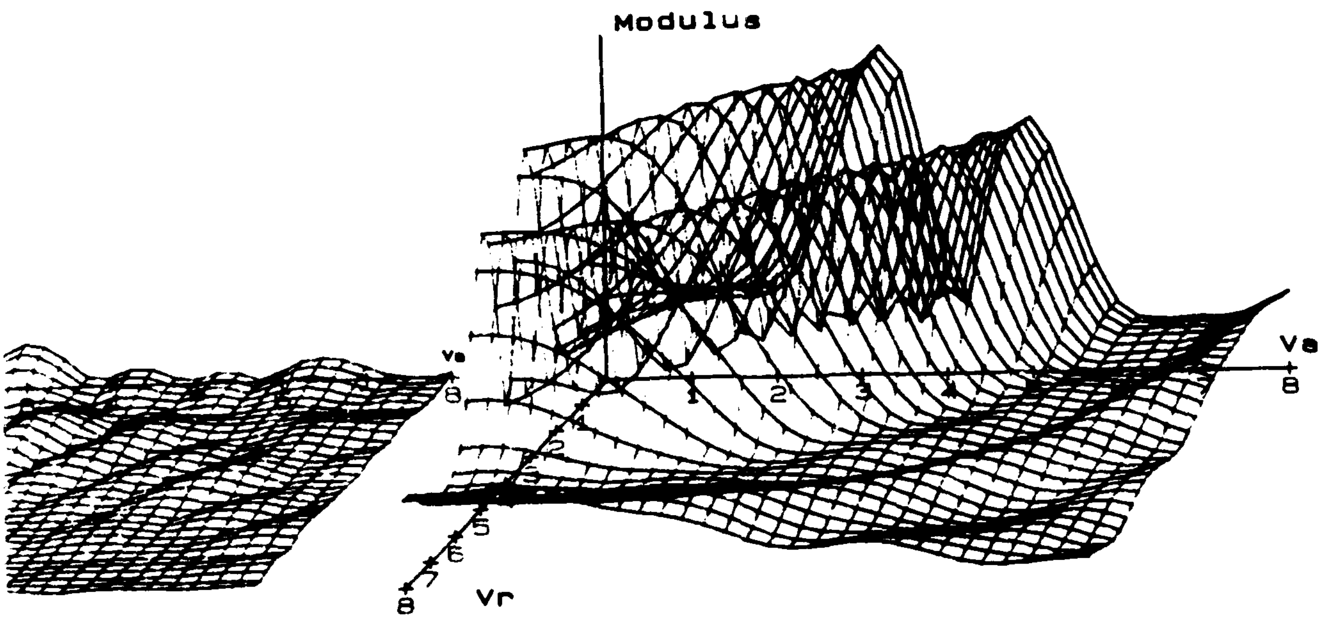



Fig 4.8. Case II Results


Modulus
Va
Vr


Modulus
Va
Vr
1
2
3
4
5
6
7
8

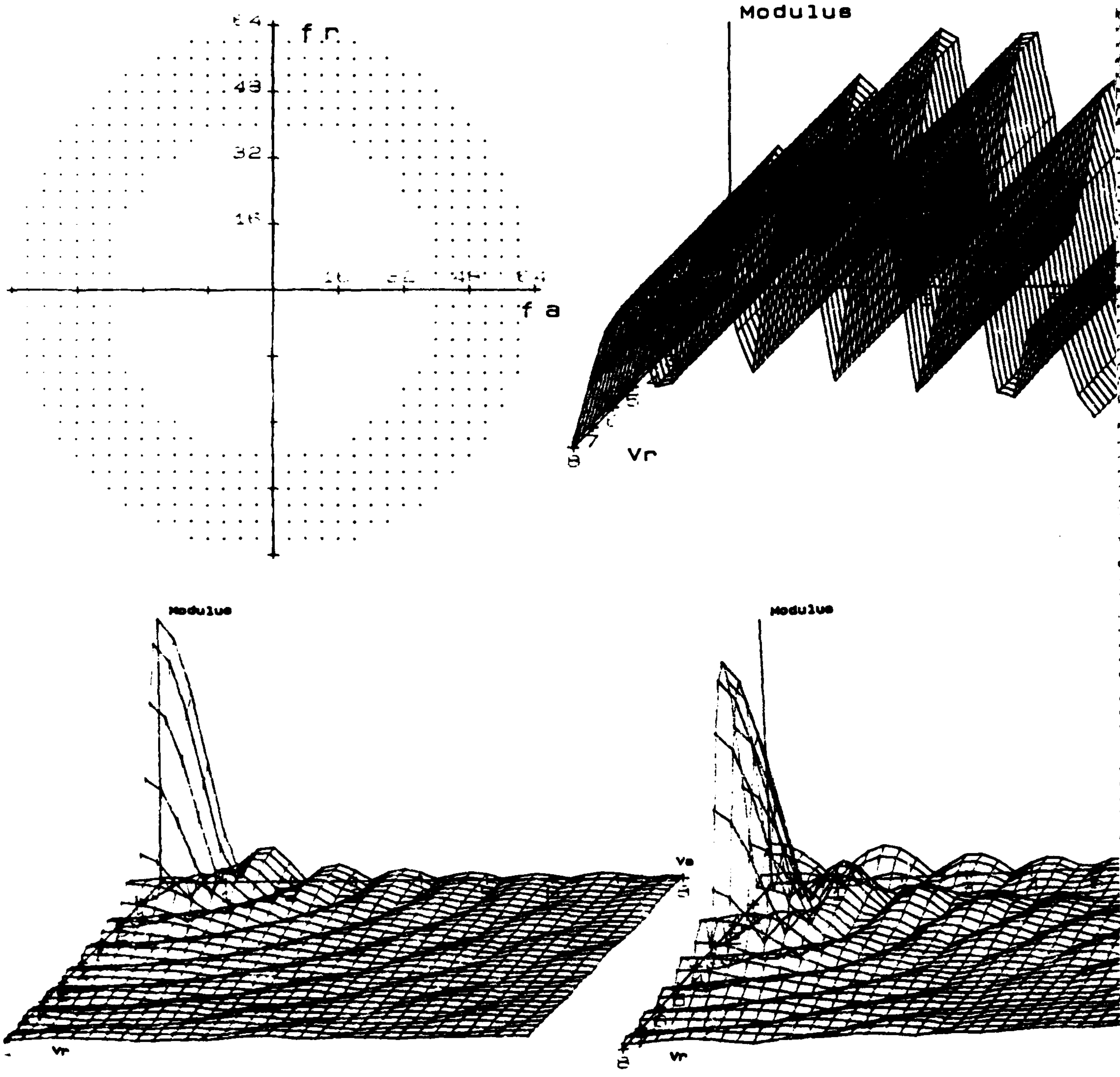
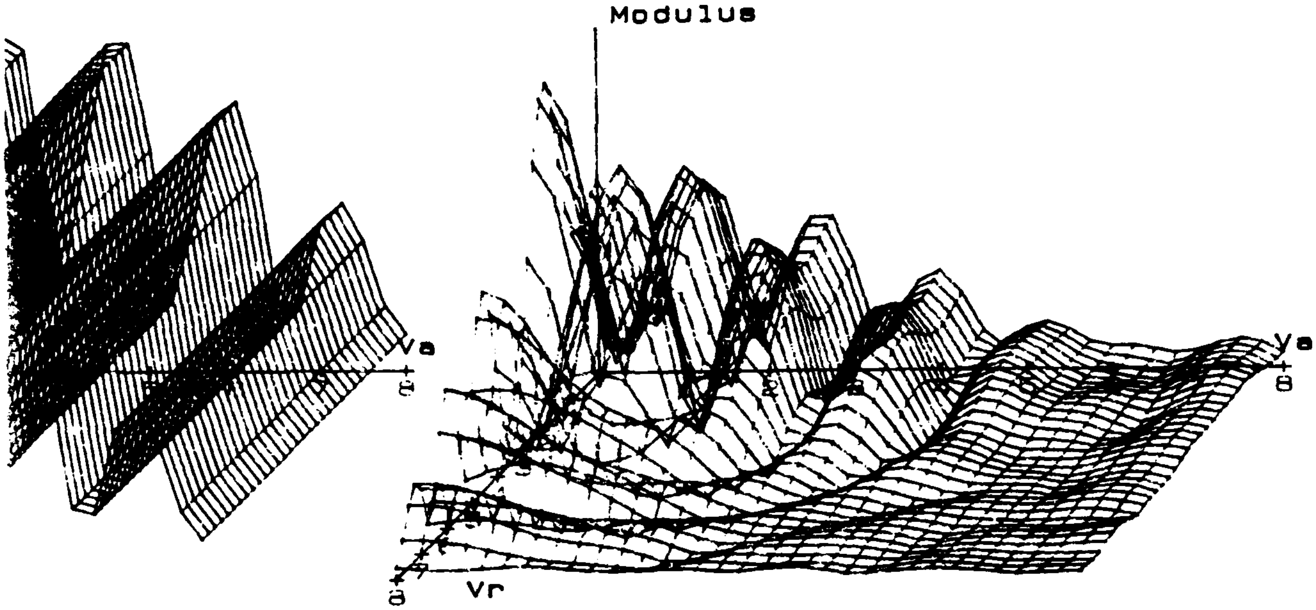
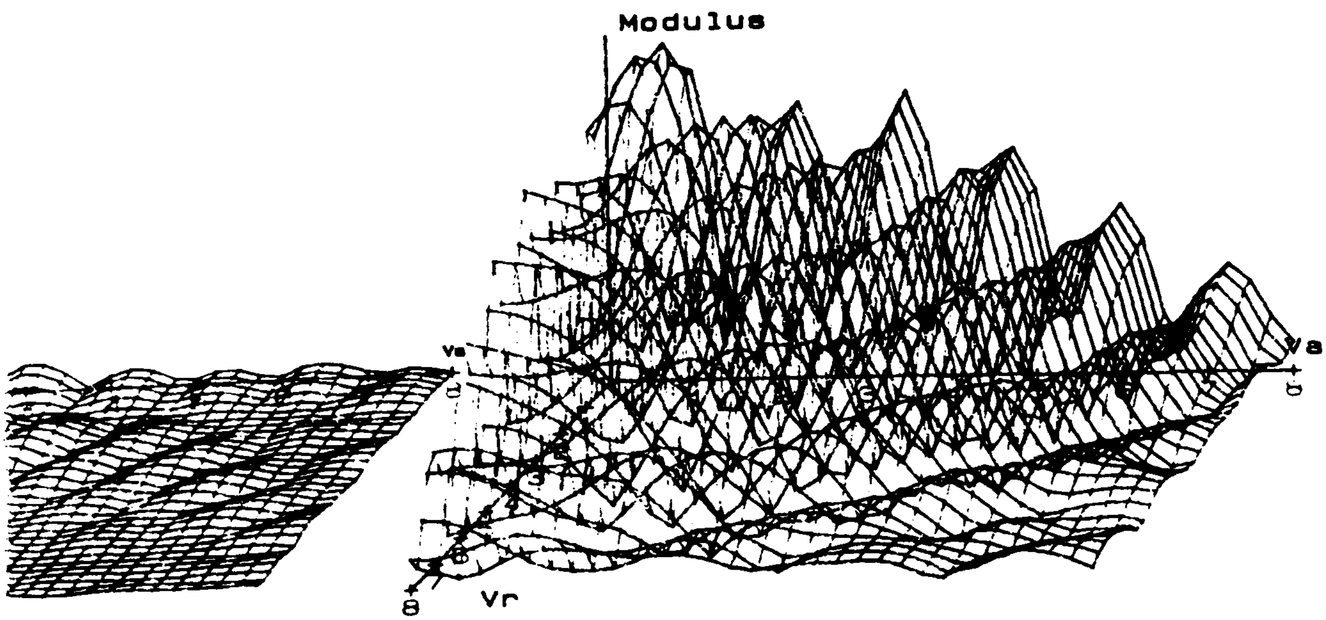



Fig 4.9. Case III Results


Modulus
Va
Vr
8


Modulus
Va
Vr
8

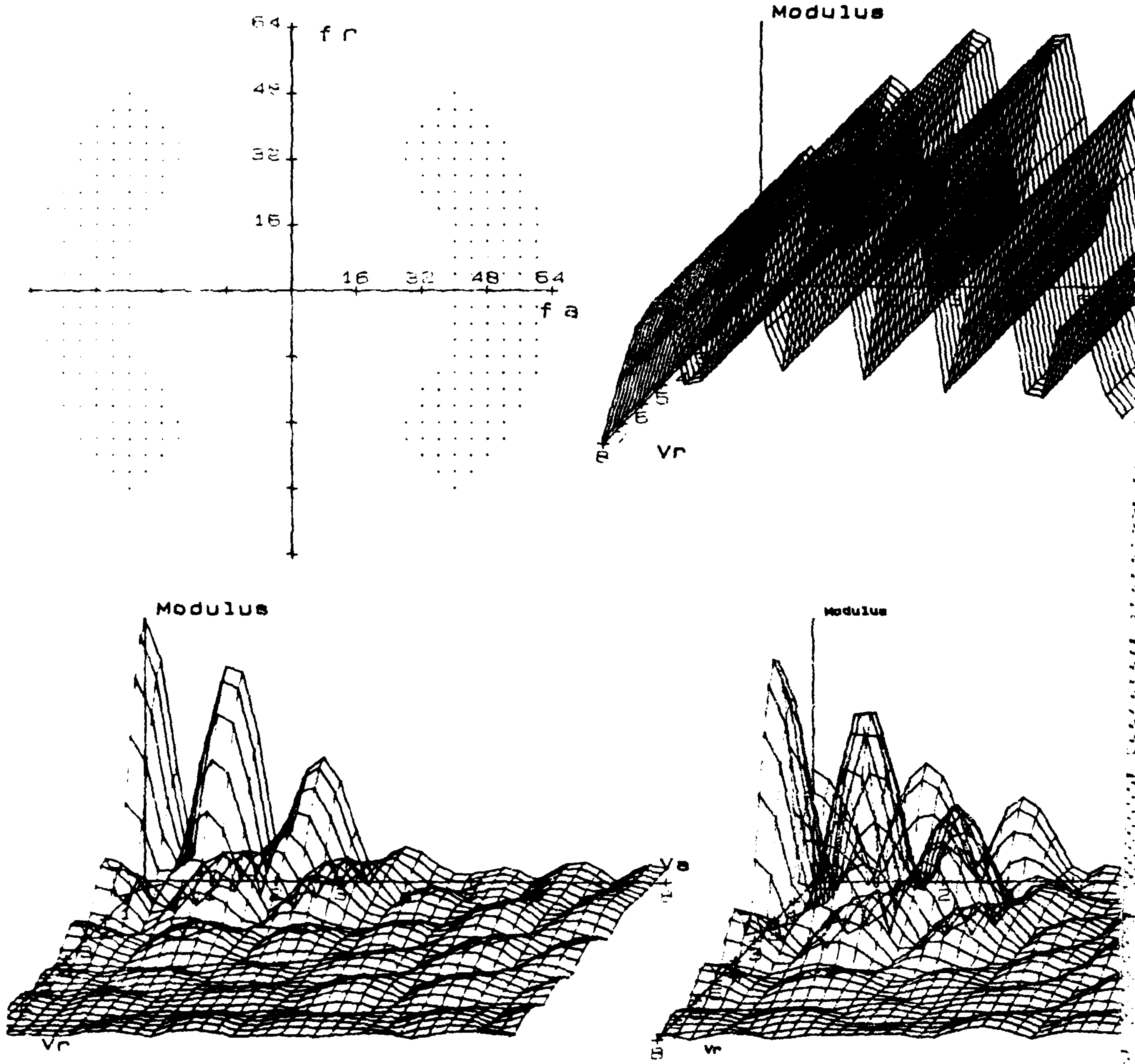
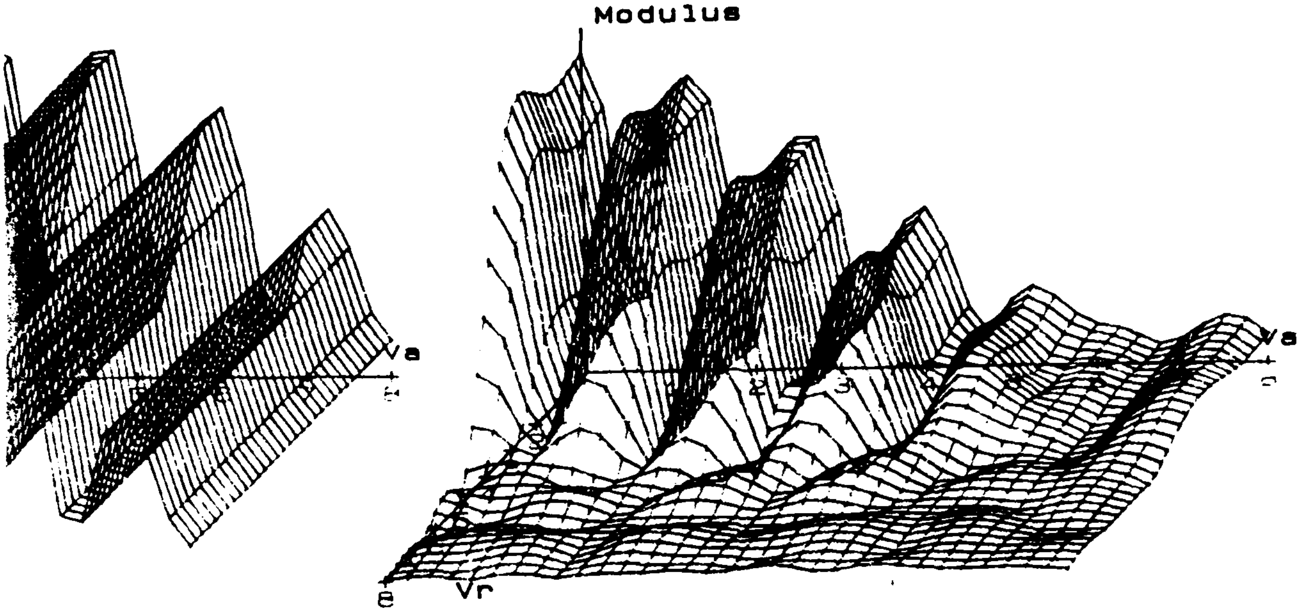
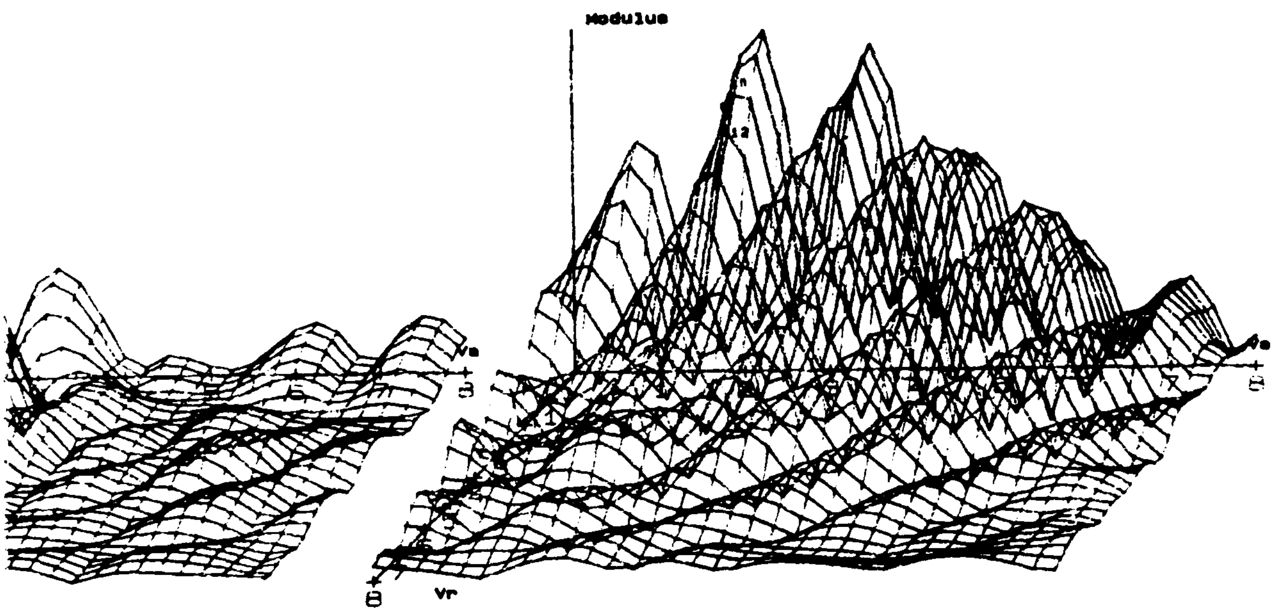



Fig 4.10. Case IV Results


Modulus
Va
Vr
8


Modulus
Va
Vr
8

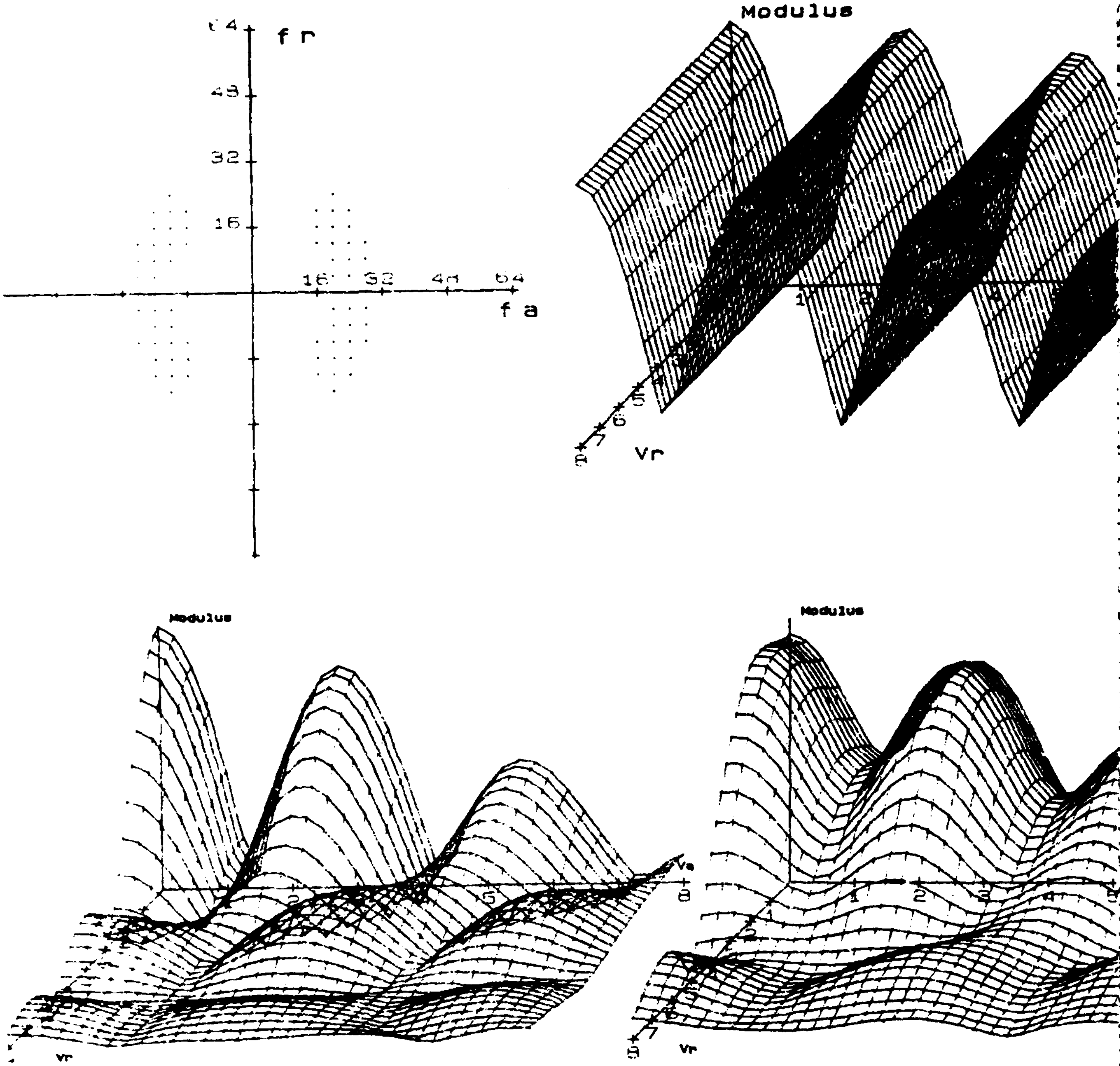
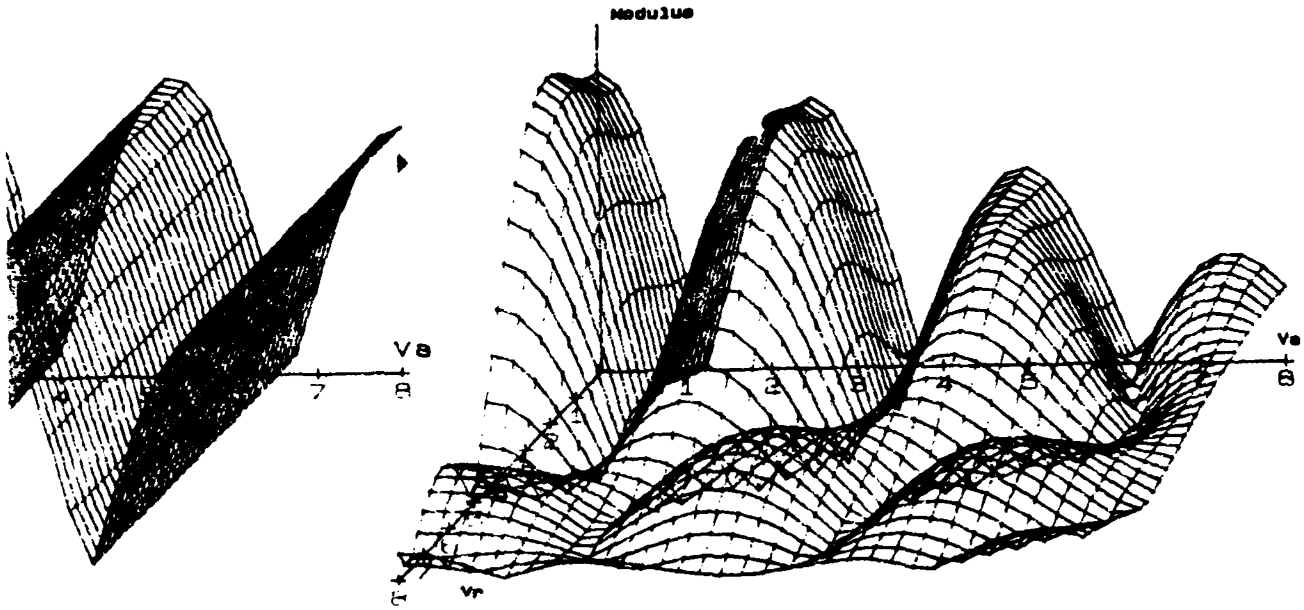
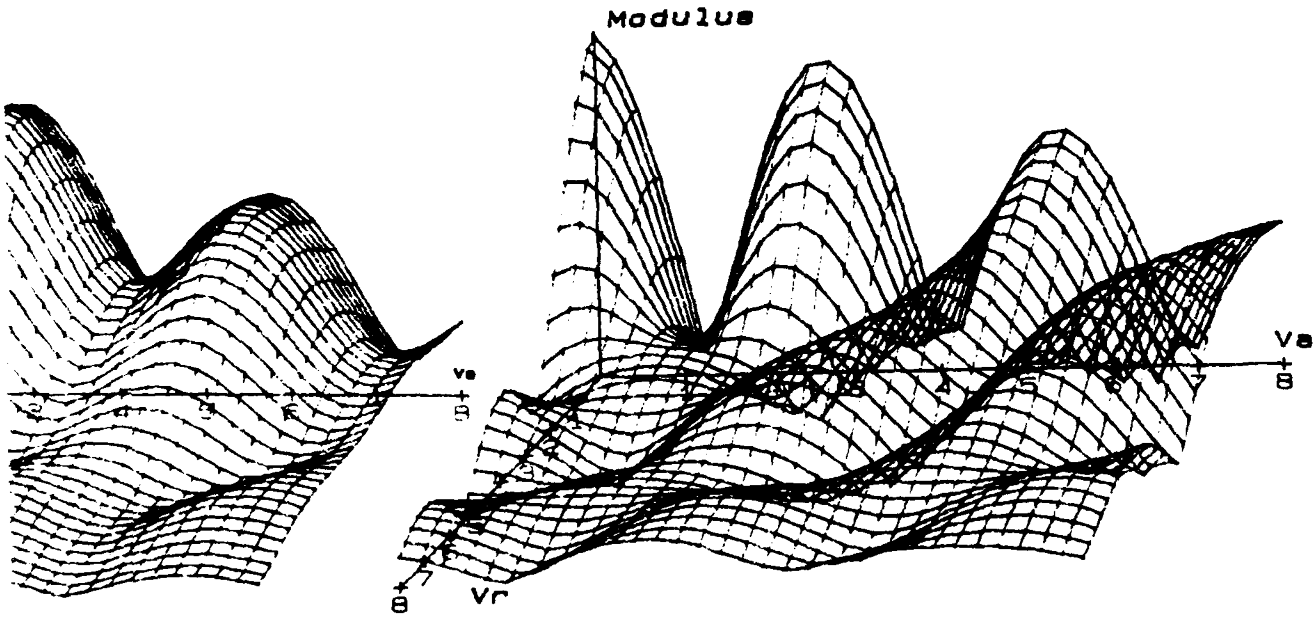



Fig 4.11. Case V Results



16,2


Modulus
Va
Va
Vr


Modulus
Va
Va
Vr

# V. CONCLUSIONS

The chapter summarizes the conclusions reached on the performance of the passive synthetic aperture system and the computer model. These conclusions are based on the results presented in the previous chapter and on the vast amount of experience gained in the actual operation of the computer model.

## Conclusions on Optical System

1. The system will behave like a high pass filter. This is because the DC component of spatial frequency will never be measured under realistic operating conditions.

2. The images will be edge enhanced. This is due to the high pass nature of the system. The images will not simply be a slightly degraded version of the geometric image but will be highly complex instead.

3. A large $\Delta\Theta$ is desirable to improve resolution along Va. Limiting $\Theta$ limits resolution along Va accordingly.

4. New methods of image interpretation will need to be developed in order for the system to be usable. The reasons cited in conclusion two will require the development of new techniques and algorithms in order to interpret the information correctly. This should be possible since the impulse response for any system configuration or change in operating conditions can be found. Knowing the impulse response of the system should enable one to find the image resulting from any input.

## Conclusions on Computer Model

1. The model performs satisfactorily. The results have been verified analytically as correct.

2. A radial FFT program would allow for the exact simulation of the system. The sampled spatial frequencies are actually distributed in a radial fashion (see Appendix A). Fitting a rectangular grid to the situation was an interim and time saving solution. A more exact simulation could be obtained through the use of radial coordinates. A program that takes an FFT in radial coordinates could not be found. Therefore, the rectangular grid was fitted in order to utilize conventional FFT programs.

3. Accuracy can be increased by increasing the array size. This will result in more samples of the input and its Fourier transform resulting in a more exact representation. This will also increase program size and slow down processing considerably.

4. Hiding lines on plots would increase the usefulness of the data. However, the method for doing this is not readily apparent.

5. The effects of phase should be determined. A phase term is present in the mutual coherence function as shown in Appendix A. Its effects have not been studied in this thesis.

# Appendix A

This appendix relates the simple case of a scene rotating beneath the lens system to the more complicated case of the lens system being carried on a collection platform and moving past the scene. This will require a rigorous derivation of the propagation of the mutual coherence function. The following paragraphs will present needed background and terminology for the derivation which follows. The material presented in this chapter is extracted mainly from reference 14:2-4 to 2-13. The system geometry is illustrated again in Figure A.1 below.

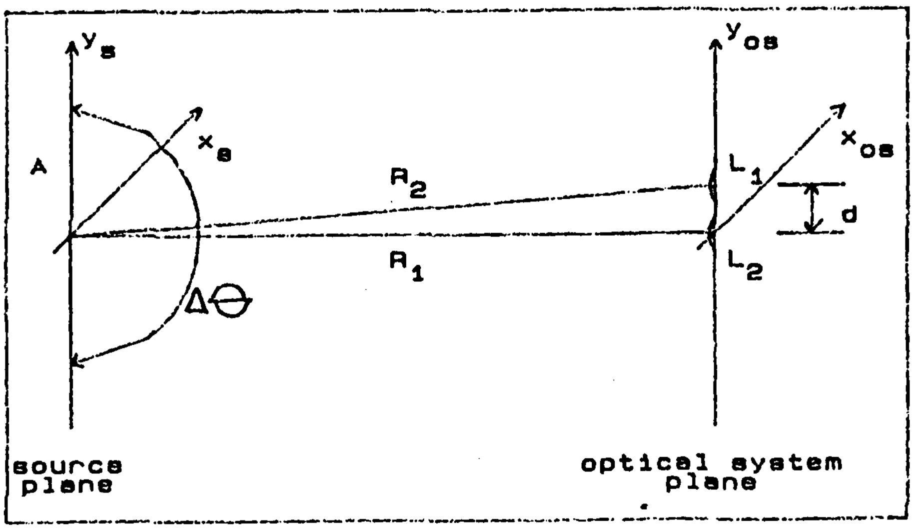



Fig A.1. System Geometry.

The major assumptions are that the measurement (slant range) plane is essentially the same as the ground plane being imaged. The lens separation is constant and the system moves in a direction parallel to this separation.

The first step is to derive the mutual coherence function at the source and next to propagate it to the lens system.

The source to be imaged has a field amplitude $E(\underline{r}',t)$ which is spatially incoherent and temporally stationary. The field at any point is uncorrelated with any other point. Therefore, the mutual coherence function $\Gamma_{12}(\tau)$ as evaluated at two points on the source is

$$\Gamma_{12}(\tau) = \langle E(\underline{r}_1',t_1)E(\underline{r}_2',t_2)^* \rangle \qquad (A.1)$$

$\Gamma_{12}(\tau)$ can be reduced as in Eq 2.6. This yields

$$\Gamma_{12}(\tau) = \langle E(\underline{r}_1',t)E(\underline{r}_2',t-\tau)^* \rangle \qquad (A.2)$$

Rewriting Eq A.2 in terms of intensity yields

$$\Gamma_{12}(\tau) = I(\underline{r}',\tau)\delta(\underline{r}_1' - \underline{r}_2') \qquad (A.3)$$

where $\delta$ is a dirac delta function denoting the spatial incoherence of the source.

Refer now to Figure A.2.

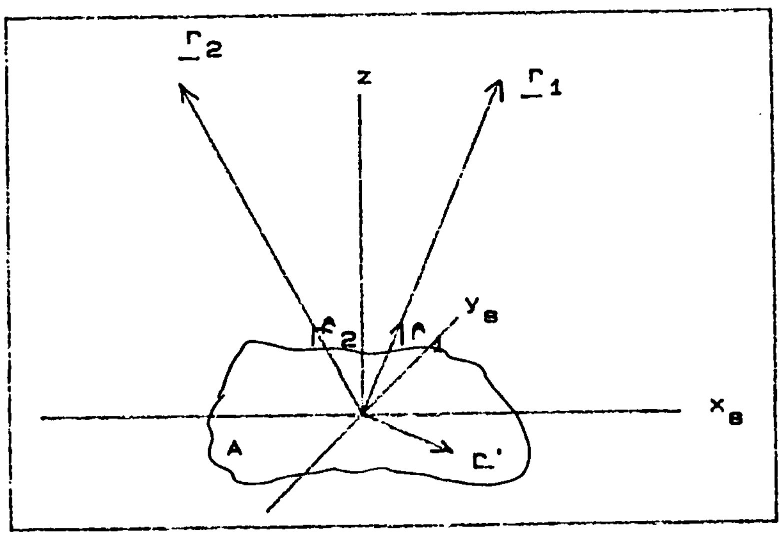



Fig A.2. Vector relationships.

The magnitude of the vectors $\underline{r}_1$ and $\underline{r}_2$ denote the distance from the center of the source to each lens, $\hat{\underline{r}}_1$ and $\hat{\underline{r}}_2$ are unit vectors along $\underline{r}_1$ and $\underline{r}_2$, and $\underline{r}'$ is the position vector of a point in the scene to be imaged. The field present at the lenses can be found through Huygens principle. This is (within a constant)

$$E(\underline{r},t) = \int_A E(\underline{r}',t - R/c)\ d\underline{r}'/R \qquad \text{(A.4)}$$

where $R = |\underline{r} - \underline{r}'|$, c is the speed of light, and R/c is the lag time from $\underline{r}'$ to $\underline{r}$.

This received signal ($s_m$) at each lens passes through a linear filter (separate but identical for each lens) with a center frequency $f_m$. This process can be denoted by

$$s_m(\underline{r},t) = E(\underline{r},t) * h_m(t) \qquad \text{(A.5)}$$

where $h_m(t)$ is the electronic impulse response of the filter and * denotes a convolution. The subscript m denotes the mth filter indicating that m total frequencies are being sampled at a given time.

A rigorous derivation of the propagation of the mutual coherence function is now made. The theoretical basis may be found in reference 2:537-599. The mutual coherence function is evaluated in terms of the received signal at each lens. This is written as

$$\Gamma(\underline{r}_1\ ,\ \underline{r}_2\ ,\ \tau) = \langle s_{m1}(\underline{r}_1',t_1)\ s_{m2}(\underline{r}_2',t_2)^* \rangle \qquad \text{(A.6)}$$

where $s_{mi}(r_i,t_i)$ is the received signal at lens i at the mth frequency received at a time $t_i$. $\Gamma_{12}$ can be rewritten by substituting for

$s_{mi}(r_i,t_i)$ as allowed by the relationship in Eq A.5. This results in

$$\Gamma_{12}(\tau) = <(E(\underline{r}_1',t_1-R_1/c)*h_m(t_1))(E(\underline{r}_2', t_2-R_2/c)*h_m(t_2))^*> \quad (A.7)$$

This equation can be rewritten once more making a substitution for $E(r_i,t_i)$ as allowed by Eq A.4. This results in

$$\Gamma_{12}(\tau) = <\int_A(( E(\underline{r}_1',t_1-R_1/c)dr'/R_1) * h_m(t_1)) \int_A((E(\underline{r}_2',t_2-R_2/c)dr'/R_2) * h_m(t_2))^* > \quad (A.8)$$

Replacing the convolution symbol with the convolution integral and removing the integrals in $\underline{r}'$ and the constants $R_1$ and $R_2$ outside the time average (since they have no time dependence) yields

$$\Gamma_{12}(\tau) = 1/(R_1R_2) \int\int_{AA} <\int_{-\infty}^{\infty}(E(\underline{r}_1',t'-R_1/c)h_m(t_1-t'))dt')\int_{-\infty}^{\infty}(E(\underline{r}_2',t''-R_2/c) h_m(t_2-t'')dt'')^*>d\underline{r}'d\underline{r}' \quad (A.9)$$

where t' and t" are dummy variables introduced by the use of the convolution integral.

Since the time average only applies to E because of its rapid oscillations, Eq A.9 can be rewritten as

$$\Gamma_{12}(\tau) = 1/(R_1R_2)\int\int\int\int_{AA-\infty}^{\infty} < (E(\underline{r}_1',t'-R_1/c)E(\underline{r}_2',t''-R_2/c) > h_m(t_1-t') h_m(t_2-t'')^* \, dt'dt'' \, d\underline{r}'d\underline{r}' \quad (A.10)$$

Employing Eq A.3 allows the quantity within the time average brackets < and > to be rewritten which yields

$$\Gamma_{12}(\tau) = 1/(R_1R_2)\int\int\int\int_{AA-\infty}^{\infty} I(\underline{r}',t'-t''-(R_1-R_2)/c)\,\delta(\underline{r}_1'-\underline{r}_2',)d\underline{r}' \; h_m(t_1-t') h_m(t_2-t'')^* \, dt'dt''d\underline{r}' \quad (A.11)$$

The delta function results in the elimination of one area integral which reduces Eq A.11 to

$$\Gamma_{12}(\tau) = 1/(R_1R_2)\iiint_{A-\infty}^{\infty} I(\underline{r}',t'-t''-(R_1-R_2)/c)h_m(t_1-t')h_m(t_2-t'')^{*}dt'\,dt''d\underline{r}' \quad (A.12)$$

The next step is to rewrite the filter impulse responses as their Fourier transforms which are denoted $H_m(f)$ (the same symbol is used for both filters since they are identical). These are (in integral form)

$$h_m(t_1-t') = \int_{-\infty}^{\infty} H_m(f)\exp(i2\pi f(t_1-t'))df \quad (A.13)$$

and

$$h_m(t_2-t'')^{*} = \int_{-\infty}^{\infty} H_m(f) * \exp(-i2\pi f(t_2-t''))df \quad (A.14)$$

The product of $H_m(f)$ and $H_m(f)^*$ may be taken outside the integrals since they are constants for all frequencies. Replacing the filter impulses in Eq A.12 with their Fourier transforms of Eqs A.13 and A.14 results in

$$\Gamma_{12}(\tau)=|H_m(f)^2)|(R_1R_2)\iiint_{A-\infty}^{\infty} I(\underline{r}',t'-t''-(R_1-R_2)/c)\int_{-\infty}^{\infty}\exp(i2\pi f(t_1-t'))df \int_{-\infty}^{\infty}\exp(-i2\pi f(t_2-t''))df\;dt'dt''d\underline{r}' \quad (A.15)$$

The integrals of the exponential functions are in the forms of delta functions. Therefore, Eq A.15 can be rewritten as

$$\Gamma_{12}(\tau) = |H_m(f)^2|/(R_1R_2)\iiint_{A-\infty}^{\infty} I(\underline{r}',t'-t''-(R_1-R_2)/c)\delta(t_1-t')\delta(t_2-t'')\;dt'dt''dr' \quad (A.16)$$

Carrying out the integrations with respect to t' and t" utilizing the sifting property of the delta function yields the following result

$$\Gamma_{12}(\tau) = (|H_m(f)|^2)/(R_1R_2)\int I(\underline{r}', t_1-t_2-(R_1-R_2)|c)\, d\underline{r}' \qquad (A.17)$$

Denoting $t_1-t_2$ as $\tau$ and rewriting I as a Fourier transform results in the following result for the mutual coherence function

$$\Gamma_{12}(\tau) = 1/(R_1R_2)\int_A\int_{-\infty}^{\infty}|H_m(f)|^2 I(\underline{r}', f)\exp(i2\pi f(\tau-(R_1-R_2)/c))\,df\,d\underline{r}' \quad (A.18)$$

$I(\underline{r}', f)$ is essentially a constant with respect to f at the frequencies of concern. $I(\underline{r}', f)$ will therefore be denoted as $I(\underline{r}')$ from this point on. It is also assumed that $H_m(f)$ is sufficiently narrowband that the factor of f in the exponent in Eq A.18 can be replaced by its center frequency value $f_m$. The actual bandwidth of the filter, B, is defined as

$$B = \int_{-\infty}^{\infty}|H_m(f)|^2\, df \qquad (A.19)$$

Lastly, a far field assumption is made; i.e.

$$R_{1,2} = \underline{r}_{1,2} - \hat{r}_{1,2}\, \underline{r}'_{1,2} \qquad (A.20)$$

where $r_{1,2}$ is the magnitude of $\underline{r}_{1,2}$.

The final form of $\Gamma_{12}(\tau)$ with these assumptions is

$$\Gamma_{12}(\tau) = C\int I(\underline{r}')\exp(-i2\pi f_m(\underline{r}_2-\underline{r}_1)\cdot\underline{r}'/c)\,d\underline{r}' \qquad (A.21)$$

where $C = (B/R_1R_2)\exp(-i2\pi f_m(\tau-(\underline{r}_2-\underline{r}_1)/c)$. The quantity to be integrated in Eq A.21 is in the form of a Fourier transform. This is denoted symbolically as

$$\Gamma_{12}(\tau) = (B/R_1R_2)\exp(-i2\pi f_m(\tau-(\underline{r}_2-\underline{r}_1)/c)\phi(f) \qquad \text{(A.22)}$$

where $\underline{f} = (f_m/c)(\underline{r}_2 - \underline{r}_1)$. If $\alpha$ is the angular separation of the lenses (and therefore the angle between $\underline{r}_1$ and $\underline{r}_2$), $\underline{f}$ can be rewritten as

$$\underline{f} = (2f_m/c)\sin(\alpha/2)\,\hat{f} \qquad \text{(A.23)}$$

where $\hat{f}$ is a unit vector in the direction of $\underline{r}_2 - \underline{r}_1$. Vector algebra shows that $\hat{f}$ is also perpendicular to the bisector between $\underline{r}_1$ and $\underline{r}_2$. The case of a rotating scene may now be considered. Figure A.3 illustrates the current situation.

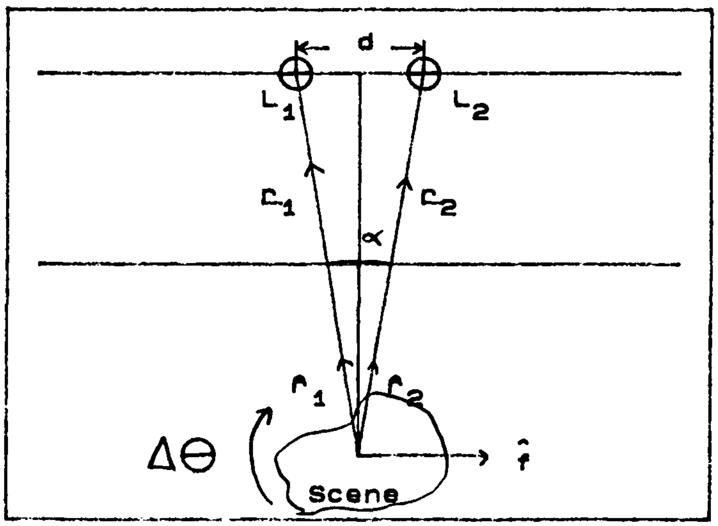



Fig A.3. Scene Rotating Beneath Lens System.

All variables are as defined in earlier chapters. Note the unit vector $\hat{f}$. Any frequency along this unit vector (denoted $\underline{f} = f_m \hat{f}$ ) can be measured by either varying the center frequency $f_m$ or by placing m linear detectors in the system. The m linear detectors would allow the simultaneous measurement of $\Gamma_{12}(\tau)$ at m different frequencies as governed by Eq A.22. The measurements are made at specific intervals as the scene rotates until a total rotation of $\Delta\theta$ = 180 degrees has been made. No more samples need be taken since the transform being sampled is hermitian (see Chapter II).

Now consider the situation in Figure A.4 where the lens system is moving while the scene remains stationary.

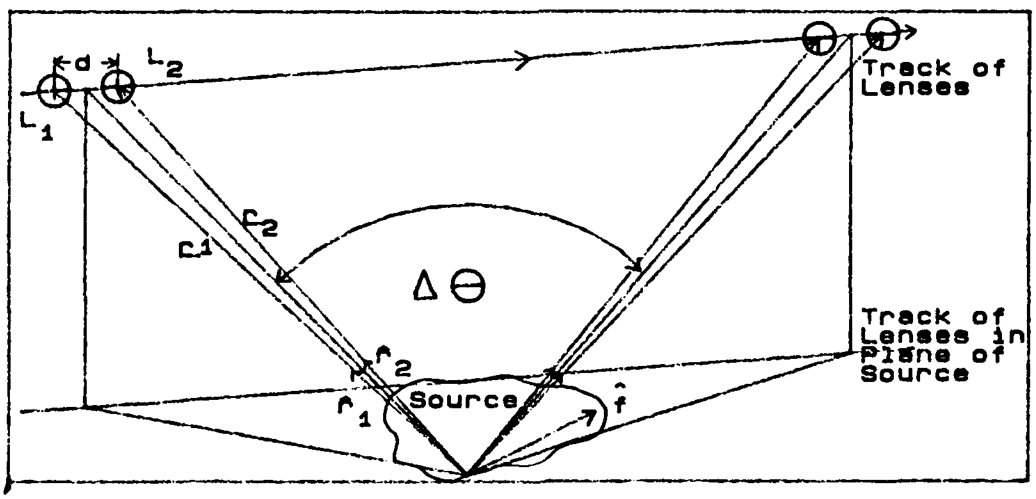



Fig A.4. Lens System Moving and Scene Stationary.

The situation is obviously identical to the rotating scene scenario. The movement of the lens system allows the unit vector $\hat{f}$ to sweep out over an angle of $\Delta\Theta$ = 180 degrees just as in the case of the rotating scene. The m frequencies can be measured as in the stationary scene case. Figure A.4 shows where the samples are taken as does Figure A.3. The frequencies measured for the four values of $\Delta\Theta$ shown are the same in both cases. A polar plot of $\Delta\Theta$ versus the spatial frequencies $f_r$ and $f_a$ for both cases yield the same results.

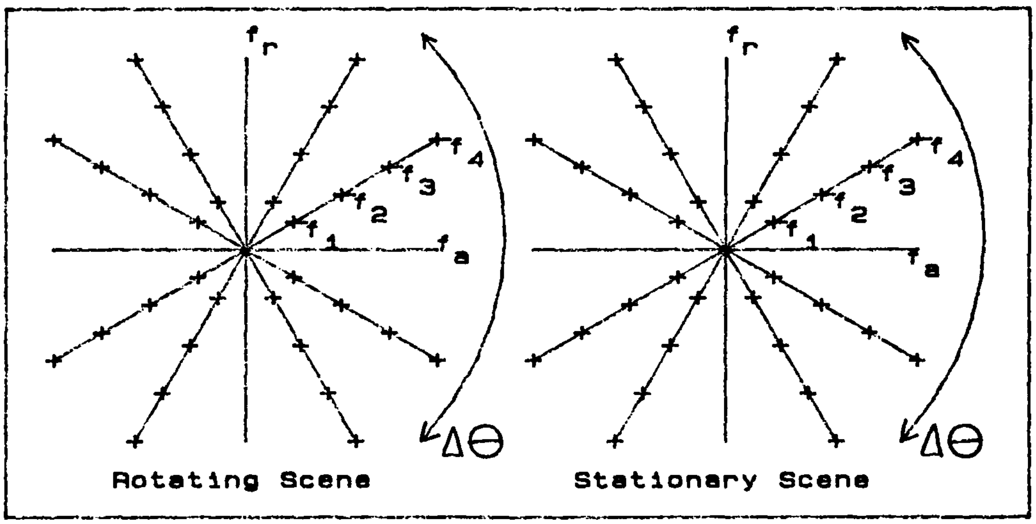



Fig A.5. Polar Plots of $\Delta\Theta$ vs $f_r$ and $f_a$ for Rotating and Stationary Cases.

This concludes appendix A. A comparison between the cases of a stationary or rotating scene was made. The results of this comparison show that both cases yield identical results. For more details, see references 2 and 14.

# Appendix B

This appendix contains the source listings of the computer model programs, and a sample model run illustrating how to use the model.

## Computer Listings

The listings presented in the following pages are in the order in which they occur when the model is run. The programs are written in Hewlett-Packard Fortran 9000 and are heavily commented.

```
      program synapt
C
C***********************************************************************
C
C     This is the main program.  The relevant data is input via
C     prompts by the program.  The program also prompts the
C     operator as to what kinds of output he desires.
C
C***********************************************************************
C
      Dimension Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 256 ) ,
     1CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) , Ar ( 2 ) , Plot ( 25
     26 , 256 )
C
C***********************************************************************
C
C     These Dimension statements declare the arrays Real, Rimag,
C     ( which are used to pass data to the FFT subroutine ), RSourc
C     and RApert ( which contain the real components of the source and
C     aperture functions respectively ) and CSourc ( which contains
C     the imaginary part of the source distribution).  The program
C     prompts the operator for the source amplitude distribution
C     and the aperture function of interest.
C
C***********************************************************************
C
      Common / Args / Real , Rimag , RSourc , CSourc , RApert
     1, Plot , Ar , Lower , Iupper , Length , Iwide
C
C***********************************************************************
C
C     This common area will be used to pass arguments more ef-
C     ficiently to the subroutines that require them.
C
C***********************************************************************
C
      Character Answer , y
      Data y/'y'/
C
C***********************************************************************
C
C     Prompt operator for type of aperture and source distributions.
C
C***********************************************************************
C
 10   Call Grphin
C
C***********************************************************************
C
C     Get the source type.  Unit 7 is the screen and unit 5 the
C     keyboard.
C
C***********************************************************************
```

```
C
      Write ( 7 , 30 )
 30   Format(/,'Enter your source irradiance distribution.  You may choo
     1se from one of the pre- programmed distributions below or create y
     2our own.  Type the number of your ',/,'selection after finding you
     3r choice on the menu below when prompted.',/,/,
     4'A point source:',T35,'1',/,
     5'A two point source:',T35,'2',/,
     6'An edge:',T35,'3',/,
     7'A slit:',T35,'4',/,
     5'A circle of variable radius:',T35,'5',/,
     6'Your own creation:',T35,'6',/,
     7/,$,'Enter your selection [1-6]: ')
C
C***********************************************************************
C
C     Call the appropriate subroutine to set the appropriate entries
C     in RSourc and CSourc to reflect the type of source desired if
C     the source chosen is not one of the five preprogrammed ones.
C
C***********************************************************************
C
      Read ( 5 , 40 ) Ichose
 40   Format(I1)
       If ( Ichose .eq  1 ) Goto 45
       If ( Ichose .eq  2 ) Call Twopnt
       If ( Ichose .eq. 3 ) Call Edge
       If ( Ichose .eq. 4 ) Call Slit
       If ( Ichose .eq. 5 ) Call Circle
       If ( Ichose .eq. 6 ) Call Other
C
C***********************************************************************
C
C     See if operator needs a picture of the source being modelled.
C
C***********************************************************************
C
 45   Write ( 7 , 50 )
 50   Format(/,$,'Do you want the source plotted on the screen [ y/n ] ?
     1 ')
      Read ( 5 , 60 ) Answer
 60   Format(A1)
      If ( Answer .ne. y ) Goto 65
      Call Pltrst ( 1 , Ichose )
 65   Write ( 7 , 70 )
 70   Format(/,$,'Do you want a hardcopy of the results [ y/n ] ? ')
      Read ( 5 , 60 ) Answer
      If ( Answer .ne. y ) Goto 75
      Call Plotin
      Call Pltrst ( 1 , Ichose )
      Call Plotof
C
C***********************************************************************
```

```
C
C      Now get the information for the aperture function ( theta ,
C      range, and lens separation ).
C
C**********************************************************************
C
 75    Call JCLR
       Call Aptinf
C
C**********************************************************************
C
C      Multiply every other element of the two functions by -1 to force
C      DC terms to the middle.
C
C**********************************************************************
C
       Call JCLR
       Call Invert ( Ichose )
C
C**********************************************************************
C
C      Transform the source.
C
C**********************************************************************
C
       Call FFTSrc ( Ichose )
       Call JCLR
C
C**********************************************************************
C
C      Multiply the transforms and pupil together.
C
C**********************************************************************
C
       Write(7,79)
 79    Format(/,'Multiplying pupil by source FFT.')
       Do 90 J = 1 , 256
          Do 80 I = 1 , 256
            RSourc ( I , J ) = RSourc ( I , J ) * RApert ( I , J )
            CSourc ( I , J ) = CSourc ( I , J ) * RApert ( I , J )
 80       Continue
 90    Continue
C
C**********************************************************************
C
C      Is a picture of this product needed?
C
C**********************************************************************
C
       Write ( 7 , 100 )
 100   Format(/,$,'Do you want a plot of the product of the FFT of the so
      1urce and the aperture',/,'distributions [ y/n ] ? ')
       Read ( 5 , 60 ) Answer
```

```
      If ( Answer .ne. y ) Goto 105
      Call Pltrst ( 4 , 0 )
 105  Write ( 7 , 110 )
 110  Format(/,$,'Do you want a hardcopy of the product [ y/n ] ? ')
      Read ( 5 , 60 ) Answer
      If ( Answer .ne. y ) Goto 115
      Call Plotin
      Call Pltrst ( 4 , 0 )
      Call Plotof
C
C***********************************************************************
C
C     Inverse FFT this resultant matrix to get source distribution.
C
C***********************************************************************
C
 115  Call JCLR
      If ( Ichose .lt. 5 ) Call Invert ( 10 )
      Call IFTSrc
      Call JCLR
C
C***********************************************************************
C
C     Would operator like to try again?
C
C***********************************************************************
C
      Write ( 7 , 120 )
 120  Format(/,$,'Do you desire to try another source and aperture [ y/n
     1 ] ? ')
      Read ( 5 , 60 ) Answer
      If ( Answer .ne. y ) Goto 140
      Do 130 J = 1 , 256
        Real ( J ) = 0.
        Rimag ( J ) = 0.
        Do 130 I = 1 , 256
          RSourc ( I , J ) = 0.
          CSourc ( I , J ) = 0.
          RApert ( I , J ) = 0.
 130  Continue
      Lower = 0
      Goto 10
 140  Call Grphof
      Stop
      End
```

0 Errors detected
204 Source lines read

```
      Subroutine Grphin
C
C***********************************************************************
C
C     This common area contains the required data.
C
C***********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C***********************************************************************
C
C     Initialize graphics terminal with AGP stuff ( see AGP graphics
C     manual ).
C
C***********************************************************************
C
      Call JBEGN
      Call JDINT ( 1 , 3 , 3Hwsp , 8 , 8H/dev/tty , 0 )
      Call JWON ( 1 )
C
C***********************************************************************
C
C     Set aspect ratio.
C
C***********************************************************************
C
      Call JIWS ( 1 , 254 , 0 , 0 , 2 , Idum , Idum , Ar )
      Call JASPK ( 1.0 , Ar ( 2 ) )
C
C***********************************************************************
C
C     Set up viewing references and rotate axes.
C
C***********************************************************************
C
      Call JVDIS ( 1.0 )
      Call JPROJ ( 0 , 0.3 , 0.3 , -1.0 )
      Wind = 1.0
      Call JWIND ( -Wind , Wind , -Wind * Ar ( 2 ) , Wind * Ar ( 2 ) )
C
C***********************************************************************
C
C     Return to calling program.
C
C***********************************************************************
C
      Return
      End
```

```
 0 Errors detected
51 Source lines read
```

```
          Subroutine Twopnt
C
C*******************************************************************************
C
C         This subroutine creates a two point source by making
C         RSourc (112,128) and RSourc (144,128) both 1.0 and leaving the
C         rest of the array 0.0.  The phase is also 0.0 at all points.
C
C*******************************************************************************
C
          Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
        16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
        2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
          Write ( 7 , 10 )
   10     Format(/,$,'Enter the separation distance ( < 128 ) as nnn : ')
          Read ( 5 , 20 ) Length
   20     Format(I3)
          Ipos = 128 - INT ( Length / 2 )
          RSourc ( Ipos , 128 ) = 1.0
          RSourc ( Ipos + Length , 128 ) = 1.0
          Return
          End
```

0 Errors detected
23 Source lines read

```
      Subroutine Edge
C
C*********************************************************************
C
C     This subroutine creates an edge as a source.  This is done by
C     first finding how wide the edge is.  The 256 rows X Iwide
C     columns are then set to 1.
C
C*********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C*********************************************************************
C
C     Find out how wide the edge is supposed to be.
C
C*********************************************************************
C
      Write ( 7 , 10 )
 10   Format(/,'How wide is the edge ( ( 128 ) ?',/,$,'Enter answer as w
     1ww : ')
      Read ( 5 , 20 ) Iwide
 20   Format(I3)
      Do 40 J = 1 , Iwide
         Do 30 I = 1 , 256
                RSourc ( I , J ) = 1.0
 30      Continue
 40   Continue
      Length = 256
      Return
      End


                  0 Errors detected
                 33 Source lines read
```

```
      Subroutine Slit
C
C*********************************************************************
C
C     This subroutine creates a slit for a source.  This is done by
C     first obtaining the length and width from the operator.
C     The middle Length and Iwide rows X columns are then set to 1.
C
C*********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C*********************************************************************
C
C     Find out the dimensions of the slit.
C
C*********************************************************************
C
      Write ( 7 , 10 )
 10   Format(/,'What is the length and width of the slit ( both < 128 )
     1?',/,$,'Enter as lll www : ')
      Read ( 5 , 20 ) Length , Iwide
 20   Format(I3,1x,I3)
C
C*********************************************************************
C
C     Center the slit around the point 128,128.  The length runs along
C     the azimuth and the width along the range.
C
C*********************************************************************
C
      Jstart = 128 - Iwide / 2
      Istart = 128 - Length / 2
      Do 40 J = Jstart , 128 + Jstart
       Do 30 I = Istart , 128 + Istart
        RSourc ( I , J ) = 1.0
        CSourc ( I , J ) = 0.
 30    Continue
 40   Continue
      Return
      End
```

0 Errors detected
43 Source lines read

```
      Subroutine Circle
C
C************************************************************************
C
C     This subroutine forms a circular source in the RSourc array.
C     The subroutine utilizes a "shading" routine in order to overcome
C     the rough edges caused by representing circle in a rectangular
C     array of points.  See comments later for details.
C
C************************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C************************************************************************
C
C     Prompt operator for radius Radius.  This is the distance out from
C     the center of the arrays that the circle will encompass.
C
C************************************************************************
C
      Write ( 7 , 10 )
 10   Format(/,$,'What is the radius of the source ( < 65 ) [ ff.ff ] ?
     1 ')
      Read ( 5 , 20 ) Radius
 20   Format(F6.2)
      Iupper = INT ( Radius )
C
C************************************************************************
C
C     The interior of the circle is now filled in.
C
C     The first do loops determine how far out from the center of the
C     array (RSourc (128,128)) should be filled with ones.  This is
C     determined to be the point just prior to being right next to
C     the outer edge of the circle.  The complex part is assumed
C     to be 0.0.
C
C************************************************************************
C
C
      Do 70 I = 0 , Iupper
         Do 30 J = 0 , Iupper - 1
           Temp = I * I + J * J
           If ( SQRT ( Temp ) .gt. Radius ) Goto 40
 30      Continue
 40      Do 50 K = 128 , 128 + J
           RSourc ( 128 + I , K ) = 1.0
           RSourc ( 128 - I , K ) = 1.0
 50      Continue
         Do 60 K = 128 , 128 - J , - 1
           RSourc ( 128 + I , K ) = 1.0
```

```
            RSourc ( 128 - I , K ) = 1.0
 60       Continue
 70   Continue
C**********************************************************************
C
C     This is where the shading begins.  The the distance to the next
C     column is still short of the radius but the column after that is
C     greater than the radius.  Shading is used here to get a more
C     accurate representation of the circle.  This is done in the
C     following manner.
C
C     The slope of a line running from the last point filled in to the
C     first point outside the circle such that the line crosses the
C     point of the radius of the circle at an amplitude of 0.5 is
C     determined.  The values of all other points outside the radius
C     is extrapolated from this slope.
C
C**********************************************************************
C
      Do 100 I = 0 , Iupper + 1
        Value = 1.
        Angle = ASIN ( Float ( I ) / Radius )
        Trad = Radius * COS ( Angle )
       Do 95 Ncols = 1 , Iupper + 4
        If ( RSourc ( 128 + I , 129 + Ncols ) .gt. 0. ) Goto 95
      Run = Abs ( Float ( Ncols - 1 ) - Trad )
      Value = Value - ( 0.5 * Run )
        If ( Value .le. 0. ) Goto 100
           RSourc ( 128 + I , 128 + Ncols ) = Value
           RSourc ( 128 - I , 128 + Ncols ) = Value
           RSourc ( 128 + I , 128 - Ncols ) = Value
           RSourc ( 128 - I , 128 - Ncols ) = Value
 95    Continue
 100  Continue
      Return
      End
```

0 Errors detected
89 Source lines read

```
      Subroutine Other
C
C*********************************************************************
C
C     This subroutine allows the operator to input a non-circular
C     distribution for the source.
C
C*********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C*********************************************************************
C
C     The operator must input the values of both the real and imagin-
C     ary parts of the source one point at a time.  The order to put
C     the points in is one row at a time.
C
C*********************************************************************
C
      Write ( 7 , 10 )
 10   Format('Input the values of your source.  You have a 256 X 256 are
     1a.  Enter values as real part, imaginary part in the format f.fff
     2fff.  Enter values by row starting at 1,1 1,2 etc.')
      Do 40 I = 1 , 256
         Do 30 J = 1 , 256
               Read ( 5 , 20 ) Rpart , Cpart
 20            Format(F5.3,1x,F5.3)
               RSourc ( I , J ) = Rpart
               CSourc ( I , J ) = Cpart
 30      Continue
 40   Continue
      Return
      End
```

0 Errors detected
35 Source lines read

```
      Subroutine Plotin
C
C*************************************************************************
C
C     This common area contains the required data.
C
C*************************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C*************************************************************************
C
C     Initialize graphics terminal with AGP stuff.
C
C*************************************************************************
C
      Call Grphof
      Call JBEGN
      Call JDINT ( 2 , 8 , 8Hwsp.7550 , 13 , 13H/dev/plt7550a , 0 )
      Call JWON ( 2 )
C
C*************************************************************************
C
C     Set aspect ratio.
C
C*************************************************************************
C
      Call JIWS ( 2 , 254 , 0 , 0 , 2 , Idum , Idum , Ar )
      Call JASPK ( 1.0 , Ar ( 2 ) )
C
C*************************************************************************
C
C     Set up viewing references and rotate axes.
C
C*************************************************************************
C
      Call JVDIS ( 1.0 )
      Call JPROJ ( 0 , 0.3 , 0.3 , -1.0 )
      Wind = 1.0
      Call JWIND ( -Wind , Wind , -Wind * Ar ( 2 ) , Wind * Ar ( 2 ) )
      Call JNEWF
C
C*************************************************************************
C
C     Return to calling program.
C
C*************************************************************************
C
      Return
      End
```

```
 0 Errors detected
52 Source lines read
```

```
      Subroutine Pltrst ( Iplot , Ichose )
C
C*******************************************************************
C
C     This subroutine plots the contents of the arrays RSourc and
C     CSourc on the HP 2623A terminal.  If the aperture function is
C     desired, RApert is plotted instead.  These arrays contain the
C     data for either the source distribution, aperture distribution,
C     their FFT, the product of their FFTs, or the final obtained
C     source distribution.  An appropriate title will appear for each
C     and is indicated by Iplot.
C
C     The following integer arrays are used to pass characters to the
C     graphics subroutine JTEXM which writes their contents to the
C     current graphics output device.  AGP requires that when arrays
C     are to be written, the data should be stored as INTEGER * 2.
C     The data in the data statements is therefore in Hollerith
C     notation for this reason.  The program manipulates data as
C     normal characters through the use of character arrays which are
C     equivalensced below to the appropriate INTEGER * 2 array.
C
C*******************************************************************
C
      Integer * 2 Const ( 8 ) , Idelta ( 7 ) , Itheta ( 7 ) , Blank ,
     1 Marker , Icount
C
C*******************************************************************
C
C     These character arrays and variables are used to plot
C     character strings to whatever device is being used to plot.
C     Cnorm contains the (in character format) normalzation constant
C     Rnorm.  Theta contains the high and low limits on the theta
C     input by the operator in Aptinf.  Inum contains the frequencies
C     that are plotted on plots of the aperture.  Dot and the numbers
C     (One, Two, etc.) are used to fill the above arrays.  Answer and
C     y are used in determining if the user wants another plot.
C
C*******************************************************************
C
      Character Cnorm ( 8 ) , Theta ( 14 ) , Inum ( 2 ) , Dot , Two ,
     1 Three , Four , Six , Eight , One , Answer , y
C
C*******************************************************************
C
C     These are the Equivalence statements referenced above.  This al-
C     lows the data to be manipulated in the program as ASCII but
C     stored in INTEGER * 2 arrays for use by the JTEXM subroutine.
C
C*******************************************************************
C
      Equivalence ( Const ( 5 ) , Cnorm ( 1 ) ) , ( Const ( 6 ) ,
     1 Cnorm ( 3 ) )  , ( Const ( 7 ) , Cnorm ( 5 ) ) , ( Const ( 8 ) ,
     2 Cnorm ( 7 ) ) , ( Itheta ( 1 ) , Theta ( 1 ) ) , ( Itheta ( 2 ) ,
```

```
     3 Theta ( 3 ) ) , ( Itheta ( 3 ) , Theta ( 5 ) ) , ( Itheta ( 4 ) ,
     4 Theta ( 7 ) ) , ( Itheta ( 5 ) , Theta ( 9 ) ) ,
     5 ( Icount , Inum ( 1 ) )
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
      Data Blank/2H  / , Dot/'.'/ , Const /2HRn,2Hor,2Hm ,2H= ,
     12H  ,2H  ,2H  ,2H  / , Idelta/2HDe,2Hlt,2Ha ,2HTh,2Het,2Ha ,2H= /
     1, Itheta/2H  ,2H  ,2Hto,2H  ,2H  ,2H D,2Heg/ , Inum/'1','6'/,
     1 Two/'2'/ , Three/'3'/ , Four/'4'/ , Six/'6'/ , Eight/'8'/ ,
     1 One/'1'/ , y/'y'/
C
C*******************************************************************
C
C     Determine plot title as follows:
C
C     Iplot = 1  Source Irradiance Distribution
C     Iplot = 2  Aperture Transmittance Function
C     Iplot = 3  FFT of Source Distribution
C     Iplot = 4  FFT of Product Source and Aperture Distribution
C     Iplot = 5  Sampled Source Distribution
C
C     Initialize Const.
C
C*******************************************************************
C
      Do 1 I = 5 , 8
        Const ( I ) = Blank
 1    Continue
      If ( Iplot .ne. 2 ) Goto 19
C
C*******************************************************************
C
C     Put Lower and Iupper into characer format for plotting.
C
C*******************************************************************
C
      Lsave = Lower
      Isave = Iupper
      Do 10 I = 1 , 3
      Itemp = Lower / ( 10 ** ( 3 - I ) )
      Ichar = Itemp + 48
      Theta ( I ) = CHAR ( Ichar )
      Lower = Lower - Itemp * ( 10 ** ( 3 - I ) )
      If ( Lower .lt. 0 ) Lower = 0
      Itemp = Iupper / ( 10 ** ( 3 - I ) )
      Ichar = Itemp + 48
      Theta ( 7 + I ) = CHAR ( Ichar )
      Iupper = Iupper - Itemp * ( 10 ** ( 3 - I ) )
      If ( Iupper .lt. 0 ) Iupper = 0
 10   Continue
```

```fortran
      Lower = Lsave
      Iupper = Isave
      Goto 85
C
C*********************************************************************
C
C     Find magnitude of input data and find the normalization
C     constant Rnorm.  This section is skipped for all source and
C     aperture plots since the data is already normalized to 1.
C
C*********************************************************************
C
 19   If ( Iplot .le. 2 ) Goto 85
      Rnorm = 0.
      Do 30 J = 128 , 194
        Do 20 I = 128 , 194
          Plot ( I , J ) = ( RSourc ( I , J ) * RSourc ( I , J ) +
     1 CSourc ( I , J ) * CSourc ( I , J ) ) * * 0.5
          If ( Plot ( I , J ) .gt. Rnorm ) Rnorm = Plot ( I , J )
 20     Continue
 30   Continue
C     Do 35 J = 128 , 194
C      Do 35 I = 128 , 194
C       Plot ( I , J ) = Plot ( I , J ) / Rnorm
C
C*********************************************************************
C
C     The following lines of code are commented out for now as per
C     Maj Mill's instructions.  This code, when executed, will put
C     the data in dB form from -100 dB to 0dB.  All data less than
C     -100 dB is stored as -100 dB.
C
C*********************************************************************
C
C       If ( Plot ( I , J ) .ne. 0. ) Then
C        Plot ( I , J ) = 20. * ALOG ( Plot ( I , J ) )
C        If ( Plot ( I , J ) .lt. - 100. ) Plot ( I , J ) = - 100.
C       Else
C        Plot ( I , J ) = - 100.
C       Endif
C35   Continue
C
C*********************************************************************
C
C     Put normalization constant in character form for writing to
C     screen.
C
C*********************************************************************
C
      Ikount = 0
      Rsave = rnorm
      Temp = Rnorm
 40   If ( Temp .lt. 1. ) Goto 50
```

```
           Temp = Temp / 10.
           Ikount = Ikount + 1
           Goto 40
   50      Ix = 1
           Rnorm = Rnorm * 10000.
           Do 80 J = 1 , Ikount + 3
            If ( Ikount .gt. 0 ) Then
             If ( J .ne. Ikount ) Goto 60
               Icon = 46
               Goto 70
             Else
              If ( J .ne. 1 ) Goto 60
               Icon = 46
               Goto 70
             Endif
   60        Icon = Rnorm / ( 10. * * ( Ikount + 4 - J ) )
             Icon = Icon + 48
   70        If ( Icon .eq. 46 ) Ix = Ix + 1
             Cnorm ( Ix ) = CHAR ( Icon )
             If ( Icon .eq. 46 ) Then
               Ix = Ix - 1
               Goto 60
             Else
               If ( Cnorm ( Ix + 1 ) .eq. Dot ) Then
                 Ix = Ix + 2
               Else
                 Ix = Ix + 1
               Endif
             Endif
             If ( Icon .ne. 46 )
         1    Rnorm = Rnorm -  ( Icon - 48 ) * ( 10. * * ( Ikount + 4 - J ) )
   80      Continue
      C
      C*********************************************************************
      C
      C     Plot the headings and axes as appropriate.
      C
      C     Ask the user if he wants 2 or three dimensional plots and tell
      C     AGP to display the data as it is called for here.
      C
      C*********************************************************************
      C
   85      Call JIVON
           Write ( 7 , 82 )
   82      Format(/,'Do you want 2 or 3 dimensional plots ?',/,$,'Enter 2 or
          13 : ' )
           Read ( 5 , 83 ) Numplo
   83      Format(I1)
      C
      C*********************************************************************
      C
      C     Choose a window size for the plots.
      C
```

```
C***********************************************************************
C
      Write ( 7 , 600 )
  600 Format(/,$,'Choose a window size (default is 1.4).  Enter as x.x :
     1 ')
      Read ( 5 , 610 ) Wind
  610 Format(F3.1)
      If ( Wind .le. 1.E-7 ) Wind = 1.4
      Call JWIND ( -Wind , Wind , -Wind * Ar ( 2 ) , Wind * Ar ( 2 ) )
C
C***********************************************************************
C
C     Set the pen color.  This is ignored for the 2623a since it is
C     only black and white.  The plotter has eight pens, seven of
C     are used in this program. They a are:
C
C     1  Black
C     2  Orange
C     3  Blue
C     4  Light Green
C     5  Dark Green
C     6  Purple
C     7  Red
C
C     All pens are 3 mm points except the red which is 7 mm.
C     JCOLR is the graphics call that changes pen color.
C
C***********************************************************************
C
      Call JCOLR ( 1 )
      Call JJUST ( 0.5 , 0.0 )
      If ( Iplot .ne. 2 ) Then
       Call J2MOV ( 0.0 , - 0.8 )
      Else
       Call J2MOV ( 0.0 , - 0.62 * Wind / 1.4 )
      Endif
C
C***********************************************************************
C
C     The above statements establish the pen color for and centers
C     the following plot headings or titles.
C
C***********************************************************************
C
       If ( Iplot .eq. 1 ) Call JTEXM ( 30 , 30HSource Irradiance Distrib
     1ution )
       If ( Iplot .eq. 2 ) Call JTEXM ( 31 , 31HAperture Transmittance Fu
     1nction )
       If ( Iplot .eq. 3 ) Call JTEXM ( 26 , 26HFFT of Source Distributio
     1n )
       If ( Iplot .eq. 4 )
     1Call JTEXM ( 33 , 33HFFT of Source X Aperture Function )
       If ( Iplot .eq. 5 ) Call JTEXM ( 27 , 27HSampled Source Distributi
```

```
     ion )
C
C***********************************************************************
C
C     If the aperture is not being plotted, print the normalization
C     constant and label the axes as appropriate.
C
C***********************************************************************
C
      If ( Iplot .ne. 2 ) Then
       Call JCOLR ( 2 )
       Call J2MOV ( 0.5 * Wind / 1.4 , 0.4 * Wind / 1.4 )
       Call JTEXM ( 9 , 9HAmplitude )
       Call J2MOV ( 0.5 * Wind / 1.4 , 0.36 * Wind / 1.4 )
       Call JTEXM ( 10 , 10HNormalized )
       Call J2MOV ( 0.4 * Wind / 1.4 , 0.32 * Wind / 1.4 )
C
C***********************************************************************
C
C     If a source is being plotted, do not print Rnorm since the data
C     is already known to be normalized by default anyways.
C
C***********************************************************************
C
       If ( Iplot .ne. 1 ) Call JTEXM ( 16 , Const )
C
C***********************************************************************
C
C     List the units appropriately for Va or Vr.
C
C***********************************************************************
C
       If ( ( Iplot .lt. 2 ) .or. ( Iplot .gt. 4 ) ) Then
        Call J2MOV ( - 0.2 * Wind / 1.4 , 0.45 * Wind / 1.4 )
        Call JCSIZ ( 0.015 , 0.05 , 1.0 )
        Call JTEXM ( 13 , 13HVr (Va) is in )
        Call J2MOV ( - 0.2 * Wind / 1.4 , 0.41 * Wind / 1.4 )
        Call JTEXM ( 9 , 9Hmultiples )
        Call J2MOV ( - 0.2 * Wind / 1.4 , 0.37 * Wind / 1.4 )
        Call JTEXM ( 13 , 13Hof 500*pi/512 )
        Call JCSIZ ( 0.035 , 0.05 , 0.0 )
       Else
C
C***********************************************************************
C
C     List the units for frequency.
C
C***********************************************************************
C
        Call J3MOV ( - 0.1 * Wind / 1.4 , 0.45 * Wind / 1.4 , 0.0 )
        Call JCSIZ ( 0.015 , 0.05 , 1.0 )
        Call JTEXM ( 13 , 13HUnits are 1/m )
        Call JCSIZ ( 0.035 , 0.05 , 0.0 )
```

```
          Endif
         Else
C
C*******************************************************************
C
C      When plotting the aperture, the delta theta and the units
C      must be listed.
C
C*******************************************************************
C
          Call JCOLR ( 1 )
          Call J3MOV ( 0.35 * Wind , 0.5 * Wind / 1.2 , 0.0 )
          Call JCSIZ ( 0.01 * Wind / 1.2 , 0.05 * Wind / 1.2 , 1.5 )
          Call JTEXM ( 14 , Idelta )
          Call J3MOV ( 0.35 * Wind , 0.45 * Wind / 1.2 , 0.0 )
          Call JTEXM ( 14 , Itheta )
          Call JCSIZ ( 0.015 * Wind / 1.2 , 0.05 * Wind / 1.2 , 0.5 )
          Call J3MOV ( - 0.5 * Wind / 1.2 , 0.5 * Wind / 1.2 , 0.0 )
          Call JTEXM ( 13 , 13HUnits are 1/m )
         Endif
C
C*******************************************************************
C
C      Select axis projection based on aperture ( two
C      dimens ons) or everything else (three dimension).  When two
C      dimensional plots are called for, the three dimensional
C      projection is still used but only amplitude and azimuth informa-
C      tion are plotted along the azimuth axis.
C
C*******************************************************************
C
  700  If ( Iplot .eq. 2 )
     1   Call JPROJ ( 0 , 0.0 , 1.0 , - 1.0 )
C
C*******************************************************************
C
C      Draw X axis.  The axis is drawn across the center of the device
C      for the aperture plots and is set 0.3 world coordinate units
C      (see AGP graphics manual for an explanation of world coordinate.
C
C*******************************************************************
C
         Call JCOLR ( 3 )
         If ( Iplot .ne. 2 ) Then
          Call J3MOV ( - 1.0 , - 0.3 , 1.0 )
         Else
          Call J3MOV ( - 1.0 , 0.0 , 0.0 )
         Endif
         Call JR3DR ( 2.0 , 0.0 , 0.0 )
         Call JJUST ( 0.0 , 0.0 )
         If ( Iplot .ne. 2 ) Then
          Call JCOLR ( 4 )
C
```

```
C***********************************************************************
C
C     Label the axes for anything but an aperture.  If plot calls for
C     units of frequency, label them each in multiples of 32.
C     If units of V are called for, label them in eight even
C     increments from 1 to 8.
C
C***********************************************************************
C
         Do 84 I = 1 , 8
          If ( Numplo .eq. 4 ) Numplo = 3
          Xpos = - 1.0 + I * 0.25
          Marker = 10 + I
          Call J3MRK ( Xpos , - 0.3 , 1.0 , 2 )
          Call J3MOV ( Xpos - 0.05 , - 0.35 , 1.0 )
          If ( ( Iplot .gt. 2 ) .and. ( Iplot .lt. 5 ) ) Then
           Goto ( 810 , 820 , 830 , 840 , 850 , 860 , 870 , 880 ) I
          Else
           Call J3MRK ( Xpos , - 0.35 , 1.0 , Marker )
           If ( Numplo .eq. 2 ) Goto 84
          Endif
 800      Ypos = 1.0 - I * 0.25
          If ( Numplo .eq. 4 ) Goto 84
          Numplo = 4
          Call J3MRK ( - 1.0 , - 0.3 , Ypos , 2 )
          Call J3MOV ( - 1.0 , - 0.35 , Ypos + 0.05 )
          If ( ( Iplot .gt. 2 ) .and. ( Iplot .lt. 5 ) ) Then
           Goto ( 810 , 820 , 830 , 840 , 850 , 860 , 870 , 880 ) I
          Else
           Call J3MRK ( - 1.0 , - 0.35 , Ypos , Marker )
           Goto 84
          Endif
 810     Call JTEXM ( 2 , 2H32 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 820     Call JTEXM ( 2 , 2H64 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 830     Call JTEXM ( 2 , 2H96 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 840     Call JTEXM ( 3 , 3H128 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 850     Call JTEXM ( 3 , 3H160 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 860     Call JTEXM ( 3 , 3H192 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 870     Call JTEXM ( 3 , 3H224 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
```

```
 880     Call JTEXM ( 3 , 3H256 )
         If ( Numplo .eq. 2 ) Goto 84
         Goto 800
 84      Continue
         If ( Numplo .eq. 4 ) Numplo = 3
C
C**********************************************************************
C
C       These statements will put a log scale on the plot if the plot
C       is desired in dBs.
C
C**********************************************************************
C
C          If ( Iplot .lt. 3 ) Goto 86
C           Call J3MOV ( - 0.2 , 0.77 , 0.0 )
C           Call JCOLR ( 6 )
C           Call JR3DR ( 0.4 , 0.0 , 0.0 )
C           Call JCOLR ( 7 )
C           Call JTEXM ( 5 , 5H 3 dB )
 86     Call J3MOV ( 0.98 , - 0.25 , 1.0 )
C
C**********************************************************************
C
C       Label the X axis fa or Va as appropriate.
C
C**********************************************************************
C
        Else
         Call J3MOV ( 0.95 , - 0.1 , 0.0 )
        Endif
       If ( ( Iplot .gt. 2 ) .and. ( Iplot .lt. 5 ) ) Then
         Call JCOLR ( 2 )
         Call J3MOV ( 0.99 , - 0.25 , 1.0 )
         Call JCOLR ( 1 )
         Call JTEXM ( 1 , 1Hf )
         Call JJUST ( 0.0 , 0.5 )
         Call JTEXM ( 1 , 1Ha )
       Else
         Call JCOLR ( 1 )
         If ( Iplot .ne. 2 ) Then
          Call JCSIZ ( 0.060 , 0.08 , 0.05 )
          Call JTEXM ( 2 , 2HVa )
         Else
          Call JCSIZ ( 0.070 , 0.10 , 0.4 )
          Call JTEXM ( 2 , 2Hfa )
          Call JCSIZ ( 0.035 , 0.05 , 0.0 )
         Endif
       Endif
C
C**********************************************************************
C
C       Draw amplitude axis if not plotting the aperture.
C
```

```
C***********************************************************************
C
      If ( Iplot .ne. 2 ) Then
       Call J3MOV ( - 1.0 , - 0.3 , 1.0 )
      Else
       Call J3MOV ( 0.0 , 0.0 , 0.0 )
      Endif
      Call JCOLR ( 3 )
      Call JR3DR ( 0.0 , 1.0 , 0.0 )
C
C***********************************************************************
C
C     Label the amplitude as being the modulus.
C
C***********************************************************************
C
      If ( Iplot .ne. 2 ) Then
       Call JCSIZ ( 0.06 , 0.08 , 0.05 )
       Call J3MOV ( - 0.95 , 0.72 , 1.0 )
       Call JTEXM ( 7 , 7HModulus )
      Else
       Continue
      Endif
C
C***********************************************************************
C
C     Draw the Y axis only for aperture and three dimensional plots.
C
C***********************************************************************
C
      If ( Iplot .ne. 2 ) Then
        Call J3MOV ( - 1.0 , - 0.3 , 1.0 )
      Else
       Call J3MOV ( 0.0 , 0.0 , - 1.0 )
      Endif
       Call JCOLR ( 3 )
      If ( Iplot .eq. 2 ) Then
       Call JR3DR ( 0.0 , 0.0 , 2.0 )
      Else
       If ( Numplo .ne. 2 ) Call JR3DR ( 0.0 , 0.0 , - 2.0 )
      Endif
      If ( Iplot .ne. 2 ) Then
       Call JJUST ( 0.0 , 0.0 )
       Call J3MOV ( - 0.85 , - 0 35 ,  - 1.0 )
      Else
       Call JJUST ( 0.0 , 0.0 )
       Call J3MOV ( 0.01 , - 0.05 , 1.0 )
      Endif
       Call JCOLR ( 1 )
C
C***********************************************************************
C
C     Label the Y axis fr or Vr as appropriate.
```

```
C
C***********************************************************************
C
      If ( ( Iplot .gt. 2 ) .and. ( Iplot .lt. 5 ) ) Then
       If ( Numplo .ne. 2 ) Then
        Call JTEXM ( 1 , 1Hf )
        Call J3MOV ( - 0.85 , - 0.36 , - 1.0 )
        Call JTEXM ( 2 , 2H r )
       Else
        Continue
       Endif
      Else
        If ( Iplot .eq. 2 ) Then
         Call JCSIZ ( 0.070 , 0.10 , 0.4 )
         Call JTEXM ( 3 , 3H fr )
         Call JCSIZ ( 0.035 , 0.05 , 0.0 )
        Else
         Call JCSIZ ( 0.060 , 0.08 , 0.05 )
         If ( Numplo .ne. 2 ) Call JTEXM ( 3 , 3H Vr )
         Call JCSIZ ( 0.035 , 0.05 , 0.0 )
        Endif
      Endif
C
C***********************************************************************
C
C     Plot the data.
C
C***********************************************************************
C
      Xincr = 1. / 128.
      Yincr = 1. / 128.
      If ( Iplot .gt. 2 ) Goto 180
      Goto ( 90 , 100 , 110 , 140 , 150 , 150 ) Ichose
C
C***********************************************************************
C
C     Plot a point source.
C
C***********************************************************************
C
 90   Call JCOLR ( 1 )
      Call J3MRK ( - 1.0 , 0.7 , 1.0 , 3 )
      Goto 500
C
C***********************************************************************
C
C     Plot a two point source.
C
C***********************************************************************
C
 100  Call JCOLR ( 1 )
      Ypos = 1.0 - ( Float ( Length ) / 2. ) * Yincr
      Call J3MRK ( - 1.0 , 0.7 , Ypos , 3 )
```

```
      Goto 500
C
C***********************************************************************
C
C     Plot an edge.
C
C***********************************************************************
C
 110  Call JCOLR ( 5 )
      Do 120 J = 1 , INT ( Iwide / 2 )
        Xpos = - 1.0 + J * Xincr
        Call J3MOV ( Xpos , 0.7 , 1.0 )
        If ( Numplo .eq. 2 ) Goto 120
        Call J3DRW ( Xpos , 0.7 , - 1.0 )
 120  Continue
      Call JCOLR ( 1 )
      Do 130 I = 1 , 256
       Ypos = 1.0 - I * Yincr
       Call J3MOV ( - 1.0 , 0.7 , Ypos )
       Call J3DRW ( Xpos , 0.7 , Ypos )
       If ( Numplo .eq. 2 ) Goto 500
 130  Continue
      Goto 500
C
C***********************************************************************
C
C     Plot a slit.
C
C***********************************************************************
C
 140  Ypos = 1.0
      Xpos = - 1.0
      Call J3MOV ( - 1.0 , 0.7 , 1.0 )
      Dist = Float ( Iwide ) * Xincr / 2.
      Call JCOLR ( 1 )
      Do 145 I = 1 , INT ( Length / 2 ) + 1
       Call JR3DR ( Dist , 0.0 , 0.0 )
       Ypos = 1.0 - I * Yincr
       If ( Numplo .eq. 2 ) Goto 500
       Call J3MOV ( Xpos , 0.7 , Ypos )
 145  Continue
      Goto 500
C
C***********************************************************************
C
C     Plot a circle.
C
C***********************************************************************
C
 150  Xincr = 1.0 / 16.
      Yincr = 1.0 / 16.
      Call JCOLR ( 1 )
      Do 170 J = 112 , 192
```

```
           Do 160 I = 112 , 192
             If ( Iplot .eq. 2 ) Then
              If ( RApert ( I , J ) .eq. 0 ) Goto 160
           Xpos = - 1.0 + ( J - 112 ) * Xincr
            Ypos = 1.0 - ( I - 112 ) * Yincr
              Call J3MRK ( Xpos , Ypos , 0.0 , 1 )
              If ( I .gt. 147 ) Goto 170
              If ( J .gt. 147 ) Goto 160
             Else
              If ( ( I .lt. 128 ) .or. ( J .lt. 128 ) ) Goto 160
              If ( RSourc ( I , J ) .eq. 0 ) Goto 160
          If ( ( Iplot .eq. 1 ) .and. ( Numplo .eq. 2 ) .and. ( I .gt. 128
        1)   ) Goto 170
              Yincr = 1 / 32.
              Xincr = 1 / 32.
              Ypos = 1.0 - ( I - 128 ) * Yincr
              Xpos = - 1.0 + ( J - 128 ) * Xincr
              Zpos = RSourc ( I , J ) - 0.3
              Call JCSIZ ( 0.070 , 0.10 , 0.4 )
              Call J3MRK ( Xpos , Zpos , Ypos , 1 )
            Endif
  160      Continue
  170     Continue
          If ( Iplot .ne. 2 ) Goto 500
           Call JCOLR ( 4 )
           Do 175 I = 1 , 4
            Pos = 0.25 * I
            If ( I .eq. 1 ) Then
             Inum ( 1 ) = One
             Inum ( 2 ) = Six
            Else
            If ( I .eq. 2 ) Then
             Inum ( 1 ) = Three
             Inum ( 2 ) = Two
            Else
             If ( I .eq. 3 ) Then
              Inum ( 1 ) = Four
              Inum ( 2 ) = Eight
            Else
             Inum ( 1 ) = Six
             Inum ( 2 ) = Four
          Endif
          Endif
          Endif
C
C**********************************************************************
C
C     Label the axes for an aperture in multiples of 16.
C
C**********************************************************************
C
              Call JCSIZ ( 0.060 , 0.08 , 0.01 )
            Call J3MRK ( Pos , 0.0 , 0.0 , 2 )
```

```
                Call J3MRK ( 0.0 , 0.0 , Pos , 2 )
                Call J3MRK ( - Pos , 0.0 , 0.0 , 2 )
                Call J3MRK ( 0.0 , 0.0 , - Pos , 2 )
                Call JJUST ( 0.5 , 0.0 )
                Call J3MOV ( Pos , 0.03 , 0.0 )
                Call JTEXM ( 2 , Icount )
                Call JJUST ( 0.0 , 0.0 )
                Call J3MOV ( - 0.15 , Pos , 0.0 )
                Call JTEXM ( 2 , Icount )
  175          Continue
              Call JPROJ ( 0 , 0.3 , 0.3 , - 1.0 )
              Goto 500
C
C*********************************************************************
C
C           Plot anything else.
C
C*********************************************************************
C
  180         Xincr = 1.0 / 32.
              Yincr = 1.0 / 32.
C             Zinc = 1 / 100.
              Call JCOLR ( 1 )
              Do 200 I = 129 , 193 , 2
              Ypos = 1.0 - ( I - 129 ) * Yincr
              Z = ( Plot ( I , 129 ) / Rsave ) - 0.3
C             Z = 1 + Plot ( I , 97 ) * Zinc - 0.3
              Call J3MOV ( - 1.0 , Z , Ypos )
               Do 190 J = 129 , 193 , 2
                Xpos = - 1.0 + ( J - 129 ) * Xincr
                Z = ( Plot ( I , J ) / Rsave ) - 0.3
C               Z = 1 + Plot ( I , J ) * Zinc - 0.3
                Call J3DRW ( Xpos , Z , Ypos )
  190          Continue
              If ( Numplo .eq. 2 ) Goto 500
  200         Continue
C
              Call JCOLR ( 5 )
              Do 220 J = 129 , 193 , 2
               Xpos = - 1.0 + ( J - 129 ) * Xincr
               Z = ( Plot ( 129 , J ) / Rsave ) - 0.3
C              Z = 1 + Plot ( 129 , J ) * Zinc - 0.3
               Call J3MOV ( Xpos , Z , 1.0 )
               Do 210 I = 129 , 193 , 2
                Ypos = 1.0 - ( I - 129 ) * Yincr
                Z = ( Plot ( I , J ) / Rsave ) - 0.3
C               Z = 1 + Plot ( I , J ) * Zinc - 0.3
                Call J3DRW ( Xpos , Z , Ypos )
  210          Continue
  220         Continue
  500         Call JIVOF
              Call JCOLR ( 1 )
              Wind = 1.0
```

```
      Call JWIND ( -Wind , Wind , -Wind * Ar ( 2 ) , Wind * Ar ( 2 ) )
      Write ( 7 , 510 )
510   Format(/,$,'Would you like to replot [ y/n ] ? ')
      Read ( 5 , 520 ) Answer
520   Format(A1)
      If ( Answer .eq. y ) Goto 85
      Return
      End
```

0 Errors detected
750 Source lines read

```
      Subroutine Plotof
C
C***********************************************************************
C
C     This subroutine turns off the graphics when the main program
C     is finished.
C
C***********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
      Call JWOFF ( 2 )
      Call JWEND ( 2 )
      Call JEND
      Call Grphin
      Return
      End
```

```
                  0 Errors detected
                 19 Source lines read
```

```
      Subroutine Aptinf
C
C***********************************************************************
C
C     This subroutine forms the aperture according to the operator's
C     desires.  The operator gives the range of theta measured.
C     The computer then determines what frequencies are sampled to
C     ensure that the samples fall exactly on a point in the array
C     and not somewhere in between.  This is done because the FFT
C     subroutine cannot work in radial coordinates.  Only frequencies
C     that lie at specific points in the aperture are measured due
C     to the bandpass limited nature of the system.
C
C     The subroutine determines the maximum theta based on the lens
C     separation ( Sep ) and range ( Range ).  These figures are input
C     by the user.  The theta that the user wishes to sample within
C     is then asked for.  The two thetas are compared to see that
C     the theta input by the user falls within the limits of the
C     system as determined by the range and separation.  The spatial
C     frequencies to be sampled are based on filter frequencies that
C     lie between 8 and 12 umicrons.  The highest spatial frequency
C     is 64 1/m.  This limit was arrived at by considering the
C     best Fourier transform vs the width of the pupil.  A pupil width
C     of 32 was found to be optimum.  64 1/m was the highest frequency
C     that could be sampled due to the bandpass of the system and
C     assuming a range of no less than 1 Km.  This frequency cor-
C     responds to the outside edge of the pupil.  Therefore, the fre-
C     quencies measured are multiples of 4 ( 4 x 16 ( radius ) = 64 ).
C
C     The subroutine determines the minimum and maximum spatial
C     frequencies based on the relationship
C
C        f =  (2 * fm / c ) * sin ( alpha / 2 ) unit vector f
C
C     where fm / c is 1 / wavelength and alpha is twice the angle
C     defined by the range from a point half way between the lenses
C     to the target and the distance from the target to a lens.
C     (See ERIM report.)  The nearest multiple of 4 is found and all
C     points within the specified theta that lie between the minimum
C     and maximum spatial frequency are set to one indicating that
C     the corresponding frequency is sampled.
C
C***********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
      Character Answer , y
      Data y/'y'/
      Pi = 3.1415926535
C
C***********************************************************************
C
```

```
C      Get the range and lens separation distance in kilometers and
C      meters respectively from the user.
C
C*********************************************************************
C
       Lower = - 1
  1    Write ( 7 , 2 )
  2    Format(/,$,'Enter range (Km) and lens separation (m) as x.x x.xx :
     1 ')
       Read ( 5 , 3 ) Range , Sep
  3    Format(F3.1,1x,F4.2)
C
C*********************************************************************
C
C      Calculate the angular separation ( in radians ) of the lenses
C      for the input range and separation.
C
C*********************************************************************
C
       Alpha2 = ATAN ( ( Sep / 2000. ) / Range )
C
C*********************************************************************
C
C      Calculate the maximum theta that can be covered assuming a
C      collector speed and stability and range specified by the user
C      ( Dtheta in degrees ).
C
C*********************************************************************
C
       Write ( 7 , 4 )
  4    Format(/,$,'Enter collector speed (ft/s) and stability as xxxx.x x
     1x.x : ')
       Read ( 5 , 5 ) Speed,Stabil
  5    Format(F6.1,1x,F4.1)
       Dtheta = 360. * ATAN ( ( Stabil * Speed * .0003048 / 2. )
     1 / Range ) / Pi
C
C*********************************************************************
C
C      Get the user's lower and upper limits on theta.
C
C*********************************************************************
C
       Write ( 7 , 9 )
  9    Format(/,'Enter theta for your particular aperture.')
 10    Write ( 7 , 20 )
 20    Format(/,'What range does theta lie within from 0 to 180 degrees?'
     1,/,/,$,'Enter your answer as nnn nnn : ')
       Read ( 5 , 30 ) Lval , Iupper
 30    Format(I3,1x,I3)
       If ( ( Lval .lt. Lower ) .or. ( Lower .eq. - 1 ) ) Lower = Lval
C
C*********************************************************************
```

```
C
C     Check to see if the theta input by the user falls within the
C     calculated theta from above.
C
C*******************************************************************
C
      If ( ( Iupper - Lval ) .le. Dtheta ) Goto 35
      Write ( 7 , 34 ) Dtheta
 34   Format(/,'You have exceeded the maximum Dtheta of ',F7.3,' degrees
     1 for the range entered.',/,'Try again.')
      Goto 1
C
C*******************************************************************
C
C     Check to see that theta lies between 0 and 180 degrees.  If not,
C     ask the operator to input theta again.
C
C*******************************************************************
C
 35   If ( ( Lval .lt. 0 ) .or. ( Lval .ge. 180 ) .or. ( Iupper .le.
     1 0 ) .or. ( Lval .ge. Iupper ) ) Then
         Write ( 7 , 40 )
 40      Format('Limits of theta are out of bounds.  Try again.')
         Goto 10
      Else
         Continue
      Endif
C
C*******************************************************************
C
C     Convert the bounds on theta to radian quantities.
C
C*******************************************************************
C
      Radlow = Lval * 3.1415927 / 180.
      Radhi = Iupper * 3.1415927 / 180.
C
C*******************************************************************
C
C     Find the maximum and minimum spatial frequencies Uprad and
C     rlorad.  All points equal to or lying between these two points
C     are set to one.  rlwav and Uwav are the lower and upper
C     wavelengths sampled by the system.
C
C*******************************************************************
C
      rlwav = 8E-6
      Uwav = 12E-6
      Uprad = ( 2. / rlwav ) * Sin ( Alpha2 )
      rlorad = ( 2. / Uwav ) * Sin ( Alpha2 )
C
C*******************************************************************
C
```

```
C      Check to see that the user has not asked for a frequency outside
C      the radius of the pupil.
C
C*********************************************************************
C
       If ( Uprad .le. 64 ) Goto 60
        Write ( 7 , 50 )
  50   Format('Your combination of lens separation and range caused',/,
      1'the upper spatial frequency to fall outside the aperture.',/,
      2'Please try again.')
        Goto 1
C
C*********************************************************************
C
C      Calculate the inner and outer radii ( squared ) between which
C      the spatial frequencies to be sampled lie.  The maximum radius
C      is rounded up and the lower radius rounded down to give the
C      largest number of frequencies sampled possible.  This gives the
C      benefit of the doubt in the system's favor.
C
C*********************************************************************
C
  60   Const = ANINT ( Uprad / 4. )
       Const = Const * Const
       Other = AINT ( rlorad / 4. )
       Other = Other * Other
C
C*********************************************************************
C
C      Now determine where the lower and upper bounds of theta lie.
C      The possibilities are 0 to 45 , 45  to 135, and 135 to 180
C      degrees.
C
C*********************************************************************
C
C*********************************************************************
C
C      0 to 45 degrees.  First compare Radlow ( the lower bound on
C      theta ) to the angles formed by the inverse tangent of the
C      column divided by the radius.  This determines whether or not
C      the section for 0 to 45 degrees should be carried out.  If
C      Radlow fall between these limits,  the program jumps to 100 to
C      determine what points in the aperture are to be set.  The exact
C      starting point is determined in the section of the program
C      starting at 100.
C
C*********************************************************************
C
       Do 70 J = 0 , 16
         Temp = SQRT ( J * J / Const )
         Angle = ATAN ( Temp )
         If ( Angle .gt. Radlow ) Goto 100
  70   Continue
```

```
C
C*******************************************************************
C
C      45 to 135 degrees.  See if lower bound on theta falls in here.
C      If Radlow was greater than 45 degrees, the subroutine continued
C      right on into this do loop.  The angle is computed as above
C      except that first pi minus the angle is checked and then pi plus
C      the angle is checked.  The subroutine then jumps to the section
C      where these angles are dealt with ( @ 150 ).
C
C*******************************************************************
C
       Do 80 I2 = 16 , - 16 , - 1
        Temp = SQRT ( I2 * I2 / Const )
        If ( I2 .ge. 0 ) Then
         Angle = ( Pi / 2. ) - ATAN ( Temp )
        Else
         Angle = ( Pi / 2. ) + ATAN ( Temp )
        Endif
         If ( Angle .gt. Radlow ) Goto 150
  80   Continue
C
C*******************************************************************
C
C      While the lower bound must lie in here by process of elimina-
C      tion ( 135 to 180 degrees ),  this check makes sure the user
C      did not goof when inputting theta.  The angle is found as
C      before and lies between 135 and 180 degrees.
C
C*******************************************************************
C
       Do 90 J2 = 16 , 0 , - 1
         Temp = SQRT ( J2 * J2 / Const )
         Angle = Pi - ATAN ( Temp )
         If ( Angle .gt. Radlow ) Goto 190
  90   Continue
       Goto 10
C
C*******************************************************************
C
C      This is where the frequencies that will be allowed to pass are
C      determined.  All the possible slopes from 0 to infinity are
C      calculated and any point lying along that slope that is not
C      outside the outer radius or less than the inner radius of the
C      pupil is set to one.  This section pertains to slopes starting
C      at 0 to 45 degs.  The angle of the slope is found by calculating
C      the inverse tangent of the column / row .
C
C      For example, the first check is to be sure that the slope's
C      angle is not less than the lower or greater than the upper
C      bound on theta.  This determines at what angle the aperture
C      will begin.  If this condition is met, fill in all points
C      along the slope greater than the square root of Other and less
```

```
C      than or equal to the square root of Const.
C      This is found by finding the magnitude of the hypotneuse of a
C      right triangle with base and height of the current row
C      and column.  The slope is the row ( Ikount ) / column ( K ).
C      This procedure is followed until the upper bound is reached.
C
C**********************************************************************
C
 100   Do 140 Ikount = 16 , 1 , - 1
         Do 130 K = 0 , 16
           If ( K .gt. Ikount ) Goto 140
           Run = FLOAT ( K )
           Rise = FLOAT ( Ikount )
C
C**********************************************************************
C
C      Get angle for this slope and be sure it does not fall outside
C      the bounds on theta.
C
C**********************************************************************
C
           Angle = ATAN ( Run / Rise )
           If ( Angle .lt. Radlow ) Goto 130
           If ( Angle .gt. Radhi ) Goto 140
C
C**********************************************************************
C
C      Determine how many columns out to "run" ( Jinc ) and how many
C      rows to "rise" ( Ikount ).  Center the circle on the point
C      128,128 of RSourc and place a one at locations where pupil is
C      to transmit.
C
C**********************************************************************
C
            Jinc = K
            Islope = Ikount
           Jrow = 0
          Do 120 I = 0 , 16 , Islope
           Rlong = I * I + Jrow * Jrow
                If ( Rlong .gt. Const ) Goto 130
           If ( Rlong .lt. Other ) Goto 115
           RApert ( 128 - I , 128 + Jrow ) = 1.0
 115       Jrow = Jrow + Jinc
 120      Continue
 130    Continue
 140   Continue
C
C**********************************************************************
C
C      Do the same as above for the rest of the pupil.
C
C**********************************************************************
C
```

```
150  Do 180 Jkount = 16 , 1 , - 1
       Do 170 K = 16 , - 16 , - 1
         If ( ABS ( K ) .gt. Jkount ) Goto 170
         Runovr = FLOAT ( K )
         Riseov = FLOAT ( Jkount )
         If ( K .ge. 0 ) Then
          Angle = ( Pi / 2. ) - ATAN ( Runovr / Riseov )
         Else
          Angle = ( Pi / 2. ) - ATAN ( Runovr / Riseov )
         Endif
         If ( Angle .lt. Radlow ) Goto 170
         If ( Angle .gt. Radhi ) Goto 180
         If ( K .eq. 0 ) Then
           Slope = 0.
         Else
           Slope =  Riseov / Runovr
         Endif
         If ( Slope .eq. 0 ) Then
          Do 155 J = 0 , 16
           J2 = J * J
           If ( ( J2 .lt. Other ) .or. ( J2 .gt. Const ) ) Goto 155
           RApert ( 128 , 128 + J ) = 1.0
155       Continue
         Else
          Irow = 0
          Iinc = - K
         Do 160 J = 0 , 16 , Jkount
          Rlong = J * J + Irow * Irow
          If ( Rlong .gt. Const ) Goto 170
          If ( Rlong .lt. Other ) Goto 156
          RApert ( 128 + Irow , 128 + J ) = 1.0
156       Irow = Irow + Iinc
160      Continue
         Endif
170    Continue
180  Continue
190  Do 230 Ikount = 16 , 1 , - 1
       Do 220 K = 16 , 0 , - 1
         If ( Ikount .lt. K ) Goto 220
         Rise = FLOAT ( Ikount )
         Run = FLOAT ( K )
         Angle = Pi - ATAN ( Run / Rise )
         If ( Angle .lt. Radlow ) Goto 220
         If ( Angle .gt. Radhi ) Goto 230
         If ( Run .eq. 0 ) Run = 0.1
         Slope = - Rise / Run
         If ( Slope .lt. - 16 ) Then
           Islope = - 1
           Jinc = 0
           Jrow = 0
         Else
           Islope = Ikount
           Jinc = K
```

```
                 Jrow = 0
               Endif
               Do 210 I = 0 , - 16 , - Islope
                Rlong = I * I + Jrow * Jrow
                If ( Rlong .gt. Const ) Goto 220
                If ( Rlong .lt. Other ) Goto 205
                RApert ( 128 - I , 128 + Jrow ) = 1.0
  205          Jrow = Jrow + Jinc
  210         Continue
  220        Continue
  230     Continue
C
C***********************************************************************
C
C         Since pupil is symmetric, fill in other half just like the other
C
C***********************************************************************
C
  240     Do 260 J = 16 , 1 , - 1
            Do 250 I = 0 , 16
              RApert ( 128 + I , 128 - J ) = RApert ( 128 + I , 128 + J )
              RApert ( 128 - I , 128 - J ) = RApert ( 128 - I , 128 + J )
  250       Continue
  260     Continue
C
C***********************************************************************
C
C         See if operator would like to add to the aperture.
C
C***********************************************************************
C
          Write ( 7 , 261 )
  261     Format(/,$,'Would you like to add to the aperture [ y/n ] ? ')
          Read ( 5 , 262 ) Answer
  262     Format(A1)
          If ( Answer .eq. y ) Goto 1
C
C***********************************************************************
C
C         See if a picture is wanted.
C
C***********************************************************************
C
          Write ( 7 , 270 )
  270     Format(/,$,'Do you want the aperture function plotted on the scree
         1n [ y/n ] ? ')
          Read ( 5 , 280 ) Answer
  280     Format(A1)
          If ( Answer .ne. y ) Goto 290
          Call Pltrst ( 2 , 5 )
  290     Write ( 7 , 300 )
  300     Format(/,$,'Do you want a hardcopy of the aperture function [ y/n
         1] ? ')
```

```
      Read ( 5 , 280 ) Answer
      If ( Answer .ne. y ) Return
      Call Plotin
      Call Pltrst ( 2 , 5 )
      Call Plotof
      Return
      End
```

```
                  0 Errors detected
                431 Source lines read
```

```
      Subroutine Invert ( Ichose )
C
C*************************************************************************
C
C     This subroutine performs magic on the source array
C     enabling the low frequency components to be displayed in the
C     center as opposed to the ends as would otherwise be the case.
C
C*************************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
C*************************************************************************
C
C     Every other element of the arrays RSourc and CSourc
C     starting with the second element is multiplied by -1.
C     The preprogrammed sources ( point, two point, edge, slit ) do
C     not need to be inverted since their transforms were calculated
C     analytically which put the low frequencies in the middle.
C
C*************************************************************************
C
      If ( Ichose .lt. 5 ) Goto 50
      Write(7,8)
 8    Format(/,'Inverting.')
      Do 20 J = 1 , 256 , 2
         Do 10 I = 2 , 256 , 2
                 RSourc ( I , J ) = - RSourc ( I , J )
                 CSourc ( I , J ) = - CSourc ( I , J )
 10      Continue
 20   Continue
C
      Do 40 J = 2 , 256 , 2
         Do 30 I = 1 , 256 , 2
                 RSourc ( I , J ) = - RSourc ( I , J )
                 CSourc ( I , J ) = - CSourc ( I , J )
 30      Continue
 40   Continue
 50   Return
      End
```

```
             0 Errors detected
            42 Source lines read
```

```
      Subroutine FFTSrc ( Ichose )
C
C************************************************************************
C
C     This program takes the 2-Dimensional FFT of the source
C     irradiance distribution.  This is done analytically for point,
C     two point, edge, and slit sources.  Other sources are done
C     using a one-dimensional fast Fourier transform (FFT) program.
C     The rows of the source are transformed first and the columns
C     second.  The transform may be displayed on the terminal and
C     plotted if desired.
C
C************************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
      Character Answer , y
      Data y/'y'/
C
C************************************************************************
C
C     Let the user know that fftsrc has been invoked.
C
C************************************************************************
C
      Write ( 7 , 5 )
 5    Format(/,'Transforming source.')
C
C************************************************************************
C
C     Ichose is passed into fftsrc from the calling program (synapt).
C     The value of Ichose corresponds to a particular type of source.
C
C     Ichose = 1 Point Source
C     Ichose = 2 Two Point Source
C     Ichose = 3 Edge Source
C     Ichose = 4 Slit Source
C     Ichose = 5 Circular Source
C     Ichose = 6 Anything Else
C
C     The computed Goto statement transfers the program to the
C     appropriate part of the program to calculate that source's
C     transform.
C
C************************************************************************
C
      Goto ( 10 , 40 , 80 , 80 , 490 , 490 ) Ichose
C
C************************************************************************
C
C     Point source has a plane wave as its transform.  Make RSourc
```

```
C      equal to 1 at all locations and CSourc equal to 0 ( phase = 0 ).
C
C*******************************************************************
C
 10    Do 30 J = 1 , 256
        Do 20 I = 1 , 256
         RSourc ( I , J ) = 1.0
         CSourc ( I , J ) = 0.0
 20     Continue
 30    Continue
       Goto 560
C
C*******************************************************************
C
C      Two point source transform is a cosine wave.  The two points
C      are separated by a distance specified by the user.  Therefore,
C      the cosine wave is a Cos ( N * pi / ( 256 / Separation ) ).
C      The columns of RSourc are set equal to twice this quantity.
C
C*******************************************************************
C
 40    Xstart = - ( Float ( Length ) / 2. ) * 3.1415926535
       Xconst = 3.1415926535 / ( 256. / Float ( Length ) )
       Do 50 I = 1 , 256
        X = Xstart + Float ( I ) * Xconst
        Real ( I ) = 2. * Cos ( X )
 50    Continue
       Do 70 J = 1 , 256
        Do 60 I = 1 , 256
         RSourc ( I , J ) = Real ( I )
 60     Continue
 70    Continue
       Goto 560
C
C*******************************************************************
C
C      Fourier transforms of an edge or a slit are similar.  The edge
C      has a finite width ( Iwide ) and an infinite length ( Length ).
C      This slit has both finite width and length.  The transform of an
C      edge is a Sinc ( pi * X / ( 256 / Iwide ) ) where 256 is the
C      periodicity of the edge.  The transform of the slit is similarly
C      calculated except that it is the product of two sinc functions.
C      The edge is oriented such that the long side runs along the
C      azimuth and its width along the range.  The slit is oriented
C      the same way.
C
C*******************************************************************
C
 90    Yconst = Float ( 256 / Iwide )
       Xconst = Float ( 256 / Length )
       Ystart = - 128. * 3.1415926535 / Yconst
       Xstart = - 128. * 3.1415926535 / Xconst
       Xincr = 3.1415926535 / Xconst
```

```
        Yincr = 3.1415926535 / Yconst
        Do 90 I = 1 , 256
         Yi = Ystart + Float ( I - 1 ) * Yincr
         If ( Abs ( Yi ) .gt. 1E-7 ) Then
          Sinc = ( Sin ( Yi ) / Yi )
         Else
          Sinc = 1.0
         Endif
         Real ( I ) = Float ( Length ) * Sinc
   90   Continue
        Do 110 J = 1 , 256
         Do 100 I = 1 , 256
          RSourc ( I , J ) = Real ( J )
  100    Continue
  110   Continue
        Do 120 I = 1 , 256
         X = Xstart + Float ( I - 1 ) * Xincr
         If ( Abs ( X ) .gt. 1E-7 ) Then
          Sinc = ( Sin ( X ) / X )
         Else
          Sinc = 1.0
         Endif
         Real ( I ) = Float ( Iwide ) * Sinc
  120   Continue
        Do 140 J = 1 , 256
         Do 130 I = 1 , 256
          RSourc ( I , J ) = RSourc ( I , J ) * Real ( I )
  130    Continue
  140   Continue
        Goto 560
C
C***********************************************************************
C
C       For anything else, calculate the FFT.  First the rows.
C
C***********************************************************************
C
  490   Do 520 I = 1 , 256
            Do 500 J = 1 , 256
                    Real ( J ) = RSourc ( I , J )
                    Rimag ( J ) = CSourc ( I , J )
  500       Continue
            Call FFT ( Real , Rimag , 256 , 1 )
            Do 510 J = 1 , 256
                    RSourc ( I , J ) = Real ( J )
                    CSourc ( I , J ) = Rimag ( J )
  510       Continue
  520   Continue
C
C***********************************************************************
C
C       Now the columns.
C
```

```
C*********************************************************************
C
      Do 550 J = 1 , 256
         Do 530 I = 1 , 256
                Real ( I ) = RSourc ( I , J )
                Rimag ( I ) = CSourc ( I , J )
530      Continue
         Call FFT ( Real , Rimag , 256 , 1 )
         Do 540 I = 1 , 256
                RSourc ( I , J ) = Real ( I )
                CSourc ( I , J ) = Rimag ( I )
 540     Continue
 550  Continue
C
C*********************************************************************
C
C     See if the transform is to be displayed or plotted.
C
C*********************************************************************
C
 560  Write ( 7 , 570 )
 570  Format(/,$,'Do you want the source FFT displayed on the screen [ y
     1/n ] ? ')
      Read ( 5 , 580 ) Answer
 580  Format(A1)
      If ( Answer .ne. y ) Goto 590
      Call Pltrst( 3 , 0 )
 590  Write ( 7 , 600 )
 600  Format (/,$,'Do you want a hardcopy of the source FFT [ y/n ] ? ')
      Read ( 5 , 580 ) Answer
      If ( Answer .ne. y ) Return
      Call Plotin
      Call Pltrst ( 3 , 0 )
      Call Plotof
      Return
      End
```

0 Errors detected
195 Source lines read

```
      Subroutine IFTSrc
C
C***********************************************************************
C
C     This subroutine calculates the inverse Fourier transform of the
C     product of the aperture and source transforms.  This product is
C     stored in RSourc and CSourc.  This inverse transform is the
C     goal of this program.  The inverse transform can be displayed on
C     the screen or a hardcopy output may be obtained.
C
C***********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
      Character Answer , y
      Data y/'y'/
C
      Write(7,29)
 29   Format(/,'Inverse transforming.')
      Do 30 I = 1 , 256
         Do 10 J = 1 , 256
                 Real ( J ) = RSourc ( I , J )
                 Rimag ( J ) = CSourc ( I , J )
 10      Continue
      Call FFT ( Real , Rimag , 256 , - 1 )
         Do 20 J = 1 , 256
                 RSourc ( I , J ) = Real ( J )
                 CSourc ( I , J ) = Rimag ( J )
 20      Continue
 30   Continue
C
      Do 60 J = 1 , 256
         Do 40 I = 1 , 256
                 Real ( I ) = RSourc ( I , J )
                 Rimag ( I ) = CSourc ( I , J )
 40      Continue
      Call FFT ( Real , Rimag , 256 , - 1 )
         Do 50 I = 1 , 256
                 RSourc ( I , J ) = Real ( I )
                 CSourc ( I , J ) = Rimag ( I )
 50      Continue
 60   Continue
      Write ( 7 , 70 )
 70   Format(/,$,'Do you want the source IFT displayed on the screen [ y
     1/n ] ? ')
      Read ( 5 , 80 ) Answer
 80   Format(A1)
      If ( Answer .ne. y ) Goto 90
      Call Pltrst ( 5 , 0 )
 90   Write ( 7 , 100 )
 100  Format(/,$,'Do you want a hardcopy of the source IFT [ y/n ] ? ')
```

```
      Read ( 5 , 80 ) Answer
      If ( Answer .ne. y ) Return
      Call Plotin
      Call Pltrst ( 5 , 0 )
      Call Plotof
      Return
      End
```

0 Errors detected
60 Source lines read

```
      Subroutine FFT ( FR , FI , N , IDIR )
C
C     Data is in FR (real) and FI (imaginary) arrays.
C     Computation is in place, output replaces input.
C     Number of points must be N.
C     FR and FI must be dimensioned in the main program.
C     IDIR = + 1 ( spatial to frequency )
C     IDIR = - 1 ( frequency to spatial )
C
      Dimension FR(1) , FI(1)
      If ( IDIR . EQ . + 1 ) goto 10
      Do 9 JJ = 1 , N
 9    FI(JJ) = -1. * FI(JJ)
 10   MR = 0
      NN = N - 1
      Do 2 M = 1 , NN
      L = N
 1    L = L / 2
      If ( MR + L .GT. NN ) goto 1
      MR = Mod ( MR , L ) + L
      If ( MR .LE. M) goto 2
      TR = FR ( M + 1 )
      FR ( M + 1 ) = FR ( MR + 1 )
      FR ( MR + 1 ) = TR
      TI = FI ( M + 1 )
      FI ( M + 1 ) = FI ( MR + 1 )
      FI ( MR + 1 ) = TI
 2    Continue
      L = 1
 3    If ( L .GE. N ) goto 12
      ISTEP = 2 * L
      EL = L
      Do 4 M = 1 , L
      A = 3.1415926535 * Float ( 1 - M ) / EL
      WR = Cos ( A )
      WI = Sin ( A )
      Do 4 I = M , N , ISTEP
      J = I + L
      TR = WR * FR ( J ) - WI * FI ( J )
      TI = WR * FI ( J ) + WI * FR ( J )
      FR ( J ) = FR ( I ) - TR
      FI ( J ) = FI ( I ) - TI
      FR ( I ) = FR ( I ) + TR
 4    FI ( I ) = FI ( I ) + TI
      L = ISTEP
      Goto 3
 12   If ( IDIR .EQ. + 1 ) Return
      Do 13 JJ = 1 , N
      FR ( JJ ) = FR ( JJ ) / Float ( N )
 13   FI ( JJ ) = -1. * FI ( JJ ) / Float ( N )
      Return
      End
```

```
 0 Errors detected
52 Source lines read
```

```
      Subroutine Grphof
C
C***********************************************************************
C
C     This subroutine turns off the graphics when the main program
C     is finished.
C
C***********************************************************************
C
      Common / Args / Real ( 256 ) , Rimag ( 256 ) , RSourc ( 256 , 25
     16 ) , CSourc ( 256 , 256 ) , RApert ( 256 , 256 ) ,
     2 Plot ( 256 , 256 ) , Ar ( 2 ) , Lower , Iupper , Length , Iwide
C
      Call JWOFF ( 1 )
      Call JWEND ( 1 )
      Call JEND
      Return
      End
```

0 Errors detected
18 Source lines read

## Sample Model Run

The following pages are taken directly from the HP 2623a graphics terminal through the use of its internal printer. Two sample runs are illustrated. The first is a flawless execution of the program to illustrate what happens when everything goes right. The second sample shows what can happen when everything goes wrong illustrating the various error messages and what the model does to recover.

Anyone wishing to run the model will first have to visit the system manager of the HP 9000 computer. He is Jeff Sweet and is located in Bldg 622 along with the computer and has an office in room 104. He can be reached by phone at extension 5-6361. He will assist the user in getting an account on the computer and getting the user familiarized with the system in general. Once logged in and in the proper account as per his instructions, the model can be invoked by typing 'synapt' followed by a carriage return. The sample model runs begin at this point.

```
/users/kane/graph [138] - synapt

Enter your source irradiance distribution.  You may choose from one of the pre-
programmed distributions below or create your own.  Type the number of your
selection after finding your choice on the menu below when prompted.

A point source:                     1
A two point source:                 2
An edge:                            3
A slit:                             4
A circle of variable radius:        5
Your own creation:                  6

Enter your selection [1-6]: 1

Do you want the source plotted on the screen [ y/n ] ? n

Do you want a hardcopy of the results [ y/n ] ? n

Enter range (Km) and lens separation (m) as x.x x.xx : 1.0 0.50

Enter collector speed (ft/s) and stability as xxxx.x xx.x : 0880.0 10.0

Enter theta for your particular aperture.

What range does theta lie within from 0 to 180 degrees?
Enter your answer as nnn nnn : 037 143

Would you like to add to the aperture [ y/n ] ? n

Do you want the aperture function plotted on the screen [ y/n ] ? n

Do you want a hardcopy of the aperture function [ y/n ] ? n

Transforming source.

Do you want the source FFT displayed on the screen [ y/n ] ? n

Do you want a hardcopy of the source FFT [ y/n ] ? n

Multiplying pupil by source FFT.

Do you want a plot of the product of the FFT of the source and the aperture
distributions [ y/n ] ? n

Do you want a hardcopy of the product [ y/n ] ? n

Inverting.

Inverse transforming.

Do you want the source IFT displayed on the screen [ y/n ] ? n

Do you want a hardcopy of the source IFT [ y/n ] ? n

Do you desire to try another source and aperture [ y/n ] ? n
/users/kane/graph [139] -
```

```
/users/kane/graph [140] - synapt

Enter your source irradiance distribution.  You may choose from one of the pre-
programmed distributions below or create your own.  Type the number of your
selection after finding your choice on the menu below when prompted.

A point source:                    1
A two point source:                2
An edge:                           3
A slit:                            4
A circle of variab1le radius:       5
Your own creation:                 6

Enter your selection [1-6]: 1

Do you want the source plotted on the screen [ y/n ] ? n

Do you want a hardcopy of the results [ y/n ] ? n

Enter range (Km) and lens separation (m) as x.x x.xx : 1.,.5

Enter collector speed (ft/s) and stability as xxxx.x xx.x : 880.,10.

Enter theta for your particular aperture.

What range does theta lie within from 0 to 180 degrees?

Enter your answer as nnn nnn : 037 143

You have exceeded the maximum Dtheta of     .000 degrees for the range entered.
Try again.


Enter range (Km) and lens separation (m) as x.x x.xx : 1.00 1.000

Enter collector speed (ft/s) and stability as xxxx.x xx.x : 888888 1

Enter theta for your particular aperture.

What range does theta lie within from 0 to 180 degrees?

Enter your answer as nnn nnn : 37 180
Your combination of lens separation and range caused
the upper spatial frequency to fall outside the aperture.
Please try again.

Enter range (Km) and lens separation (m) as x.x x.xx :
```

## BIBLIOGRAPHY


1. Beran, M. J. and G. B. Parrent, Jr. Theory of Partial Coherence. Englewood Cliffs, NJ: Prentice Hall, Inc., 1964.

2. Born, Max and Emil Wolf. Principles of Optics. Oxford, NY: Pergamon Press, 1975.

3. Bracewell, R. N. "Radio Astronomy of Discrete Sources," Proceedings of the IRE, 46: 97-105 (January 1958).

4. Environmental Research Institute of Michigan, "Passive Coherence Experiment," Technical Proposal, I: 2-4 to 2-13 (June 1984).

5. Fomalont, E. B. "Earth Rotation Aperture Synthesis," Proceedings of the IEEE, 61: 1211-1218 (September 1973).

6. Gaskill, Jack D. Linear Systems, Fourier Transforms, and Optics. New York: John Wiley & Sons, 1978.

7. Goodman, Joseph W. Introduction to Fourier Optics. San Francisco: McGraw-Hill Book Co., 1968.

8. Gush, H.P. "Optical Imaging Using Aperture Synthesis," Journal of the Optical Society of America, 69 (1): 187-191 (January 1979).

9. Ko, H. C. "Coherence Theory of Radio Astronomical Measurements, "IEEE Transactions on Antennas and Propagation, 15: 10-20 (January 1967).

10. O'Neill, Edward L. "Spatial Filtering in Optics," IRE Transactions on Information Theory, 2 : 56-65 (June 1956).

11. Oppenheim, A. V. and others. "Signal Synthesis and Reconstruction from Partial Fourier Domain Information," Journal of the Optical Society of America, 73 (11): 1413-1420 (November 1983).

12. Sato, T. and others. "Computer Controlled Image Sensor and Its Application," Applied Optics, 18 (4): 485-488 (February 1979).

13. Swenson, G. W. Jr. and N. C. Mathus. "The Interferometer in Radio Astronomy," Proceedings of the IEEE, 56 (12): 2114-2129 (December 1968).

14. Tai, Anthony M. and Carl C. Aleksoff. "An Incoherent Optical Processor for Real Time Complex Fourier Transformation, "International Computing Conference (10th), 422: 99-104 (April 1983).

15. Wolf, E. Progress in Optics. New York: American Elsevar Publishing Co., Inc., 1969.

16. Barakat, R. "Diffraction Images of Coherntly Illuminated Objects in the Presence of Aberrations", Optics Acta, 16 (2): 205-223 (March 1969).

# VITA

Captain Christopher P. Kane was born on 26 January 1959 in Oregon, Ohio. He graduated from Clay High School in 1977 and attended the University of Notre Dame from which he received the degree of Bachelor of Science in Electrical Engineering in May 1981. Upon graduation, he received a commission in the USAF through the ROTC program. He was called to active duty in June 1981 and served with the 544 Intelligence Exploitation Squadron, Offutt AFB, Nebraska where he served as an Electronic Intelligence Analysis Engineer. He served in this capacity until entering the School of Engineering, Air Force Institute of Technology, in May 1984.

Permanent Address: 435 Sylvandale
Oregon, Ohio 43616

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

# REPORT DOCUMENTATION PAGE

**1a. REPORT SECURITY CLASSIFICATION:** UNCLASSIFIED

**1b. RESTRICTIVE MARKINGS:**

**2a. SECURITY CLASSIFICATION AUTHORITY:**

**2b. DECLASSIFICATION/DOWNGRADING SCHEDULE:**

**3. DISTRIBUTION/AVAILABILITY OF REPORT:** Approved for public release; distribution unlimited

**4. PERFORMING ORGANIZATION REPORT NUMBER(S):** AFIT/GEO/ENP/85D-2

**5. MONITORING ORGANIZATION REPORT NUMBER(S):**

**6a. NAME OF PERFORMING ORGANIZATION:** School of Engineering, Air Force Institute of Technology

**6b. OFFICE SYMBOL (If applicable):** AFIT/ENP

**7a. NAME OF MONITORING ORGANIZATION:**

**6c. ADDRESS (City, State and ZIP Code):** Wright-Patterson AFB, Ohio 45433

**7b. ADDRESS (City, State and ZIP Code):**

**8a. NAME OF FUNDING/SPONSORING ORGANIZATION:** AFWAL/AARI, WPAFB, OH 45433-6543

**8b. OFFICE SYMBOL (If applicable):** AFWAL/AARI

**9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER:**

**8c. ADDRESS (City, State and ZIP Code):**

**10. SOURCE OF FUNDING NOS.**

| PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT NO. |
|---|---|---|---|
| | | | |

**11. TITLE (Include Security Classification):** See Box 19

**12. PERSONAL AUTHOR(S):** Christopher P. Kane, Capt, USAF

**13a. TYPE OF REPORT:** MS Thesis

**13b. TIME COVERED:** FROM ______ TO ______

**14. DATE OF REPORT (Yr., Mo., Day):** December 1985

**15. PAGE COUNT:** 122

**16. SUPPLEMENTARY NOTATION:**

**17. COSATI CODES**

| FIELD | GROUP | SUB. GR. |
|---|---|---|
| 17 | 05 | |
| | | |

**18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number):** Passive, Synthetic Aperture, van Cittert-Zernike, Imaging System

**19. ABSTRACT (Continue on reverse if necessary and identify by block number)**

Title: COMPUTER MODEL OF A PASSIVE SYNTHETIC APERTURE IMAGING SYSTEM

Thesis Chairman: James P. Mills, Major, USAF
Assistant Professor of Physics

Approved for public release: IAW AFR 190-1.
Lynn E. Wolaver 16 JAN 86
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AFC)
Wright-Patterson AFB OH 45433

**20. DISTRIBUTION/AVAILABILITY OF ABSTRACT:** UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐

**21. ABSTRACT SECURITY CLASSIFICATION:** UNCLASSIFIED

**22a. NAME OF RESPONSIBLE INDIVIDUAL:** James P. Mills, Major, USAF

**22b. TELEPHONE NUMBER (Include Area Code):** 513-255-2012

**22c. OFFICE SYMBOL:** AFIT/ENP

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

This thesis was concerned with the development of a computer model of a passive synthetic aperture imaging system. The research was divided into three parts. They were (1) applying an understanding of partial coherence theory and its relationship to the impulse response of the system, (2) developing the computer model, and (3) exercising the computer model to perform a sensitivity analysis.

The system modeled consisted of two lenses mounted on a movable platform. The lenses were separated by a fixed distance and travelled in a direction parallel to this separation. The coherence of radiation present at each lens emanating from a real source was measured yielding the Fourier transform of the source intensity distribution according to the van Cittert-Zernike theorem (2:510). The transform was then multiplied by an effective aperture (obtained from the motion and position of the lenses relative to the source). An inverse Fourier transform was then applied to this result yielding the image. This is the process modeled by the computer.

The results indicated that new means of image interpretation must be developed in order to make the results useful. This is due to the fact that the system behaves much like a high pass filter and the image is edge enhanced and not a scaled version of the geometric image.